Tire Pressure Monitor Sensor Product Specification

The MPXY8300 Series is a 20-pin sensor for use in applications that monitor tire pressure and temperature. It contains the pressure, temperature and 2-axis accelerometer sensors, a microcontroller, and an RF output all within a single package.

The MPXY8300 Series is comprised of the following functions all within the same package.

Features

- Pressure sensor with signal interface to ADC10
- Temperature sensor with signal interface to ADC10
- Z- and X-axis accelerometers with signal interface to ADC10
- Voltage reference measured by ADC10
- 4-channel, 10-bit analog-to-digital converter
- 8-bit MCU
    - S08 Core with SIM, interrupt and debug/monitor
    - 512 RAM
    - 8K FLASH (in addition to 8K providing factory firmware and trim data)
    - 32-byte, low power, parameter registers
- Internal 315/434 MHz RF transmitter
    - External crystal oscillator
    - PLL-based output with fractional-n divider
    - ASK and FSK modulation capability
    - Programmable data rate generator
    - Manchester or bi-phase data encoding
    - 128-bit RF data buffer with RTS/CTS handshake
    - Direct access to RF transmitter from MCU for unique formats
    - Supply voltage charge pump to compensate for cold batteries with selectable charge times (30, 60, 120 and 240 msec)
- Differential input LF detector/decoder
- 4 GPIO pins with optional pull-ups/pull-downs and wake-up interrupt
- Real Time Interrupt driven by LFO with interrupt intervals of 8, 16, 32, 64, 128, 256, 512 or 1024 msec
- Low power wake-up timer and periodic reset driven by LFO
- Watchdog time-out with selectable times and clock sources
- 2-channel general purpose timer/PWM module (TPM1)
- Internal oscillators
    - MCU bus clock of 0.5, 1, 2 and 4 MHz (1, 2, 4 and 8 MHz HFO)
    - Charge pump clock of 10 MHz
    - Low frequency, low power time clock (LFO) with 1 msec period
    - Medium frequency, LFR decoder and sensor clock (MFO) of 8 μsec period
- Low voltage detection
- Normal temperature restart in hardware (over temperature detected by software)

MPXY8300 Series

TIRE PRESSURE, TEMPERATURE AND ACCELERATION SENSOR
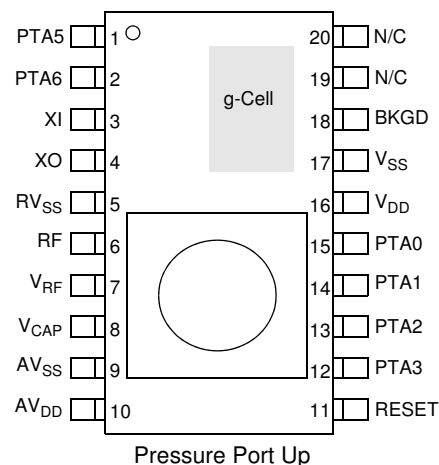

20-LEAD
SOIC
CASE 1793-03

Top View

PTA5	1
PTA6	2
XI	3
XO	4
RV$_{SS}$	5
RF	6
V$_{RF}$	7
V$_{CAP}$	8
AV$_{SS}$	9
AV$_{DD}$	10

g-Cell

20	N/C
19	N/C
18	BKGD
17	V$_{SS}$
16	V$_{DD}$
15	PTA0
14	PTA1
13	PTA2
12	PTA3
11	RESET

Pressure Port Up

Figure 1. Pin Connections

freescale™
semiconductor

ORDERING INFORMATION			
MPXY8300 Series Order Numbers	Temperature Range	Case No.	Package
MPXY8300A6T1	-40°C to 125°C	1793-03	SOIC-20, Tape and Reel
MPXY8300A6U	-40°C to 125°C	1793-03	SOIC-20, Rail
MPXY8300B6T1	-40°C to 125°C	1793-03	SOIC-20, Tape and Reel
MPXY8300B6U	-40°C to 125°C	1793-03	SOIC-20, Rail
MPXY8300C6T1	-40°C to 125°C	1793-03	SOIC-20, Tape and Reel
MPXY8300C6U	-40°C to 125°C	1793-03	SOIC-20, Rail

SECTION 1 GENERAL DESCRIPTION

1.1 Overall Block Diagram

The block diagram of the device is shown in Figure 1-1. This diagram covers all the main blocks mentioned above and their main signal interactions. Power management controls and bus control signals are not shown in this block diagram for clarity.

1.2 Multi-Chip Interface

The device contains four devices using the best process technology for each.
- Microcontroller (MCU)
- RF transmitter device (RFX)
- Pressure sensing device (P-CHIP)
- Z- and X-axis acceleration transducer (g-Cell)

As shown in Figure 1-1 the MCU interfaces to the RFX using a standard serial peripheral interface (SPI) which creates a separate data and address bus structure within the RFX. The P-Chip and g-Cell both interface to the MCU.

1.3 System Clock Distribution

The various clock sources and their distribution are shown in Figure 1-2. Note that most clock sources and distribution are limited to one device, while the SFO, D_X and SPI clocks are connected between devices. All clock sources except the low frequency oscillator, LFO, can be turned off by software control in order to conserve power. The LFO is used for:
- Slow periodic wake-up of the MCU (see Section 11)
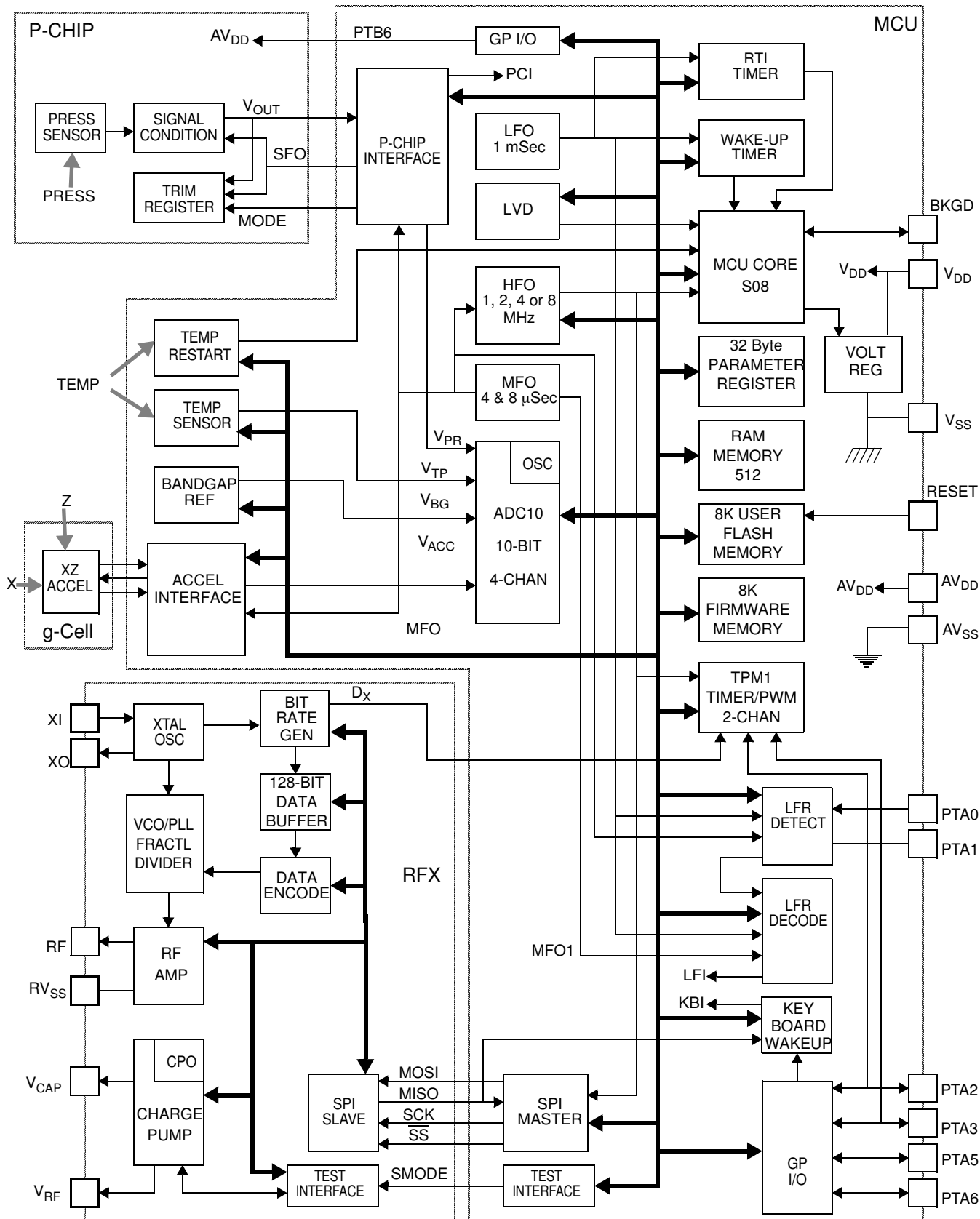- Slow periodic wake-up of LF receiver (LFR) (see Section 12)

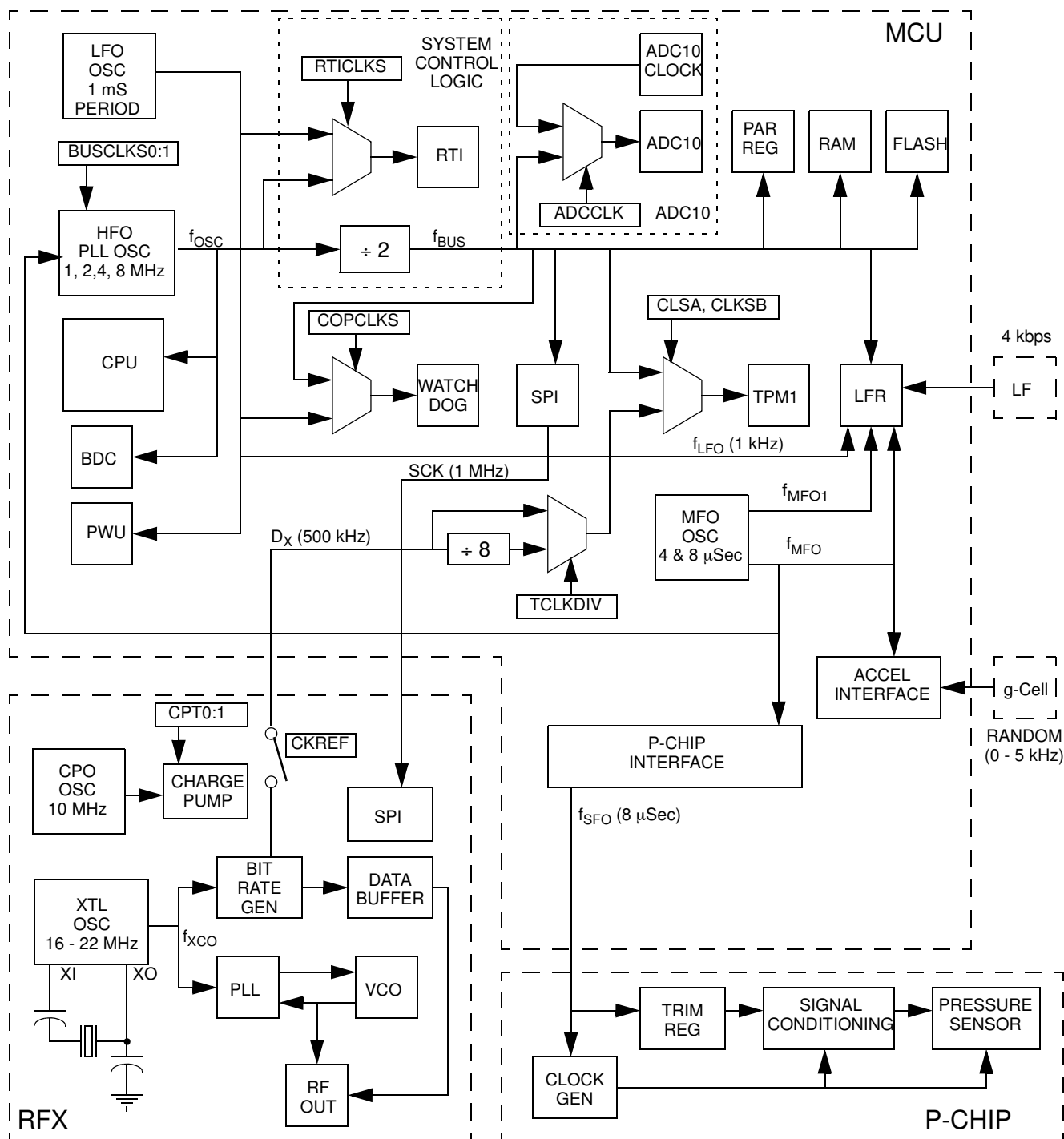**Figure 1-1 MPXY8300 Series Overall Block Diagram**

Figure 1-2 MPXY8300 Series Clock Distribution

1.4   Reference Documents

The MPXY8300 Series utilizes a custom microcontroller. The user can obtain further detail on the full capabilities of the MCU modules by referring to the HCS08 Family Reference Manual (HCS08RMV1).

SECTION 2 PINS AND CONNECTIONS

This section describes the pin layout and general function of each pin.

2.1   Package Pinout

The pinout for the 20-pin MPXY8300 Series package is shown in Figure 2-1 for the pressure port up orientation as shown in the cross sections in Figure 2-2. Both orientations use the standard method for the order of pin numbers, but the order of the pin functions changes.
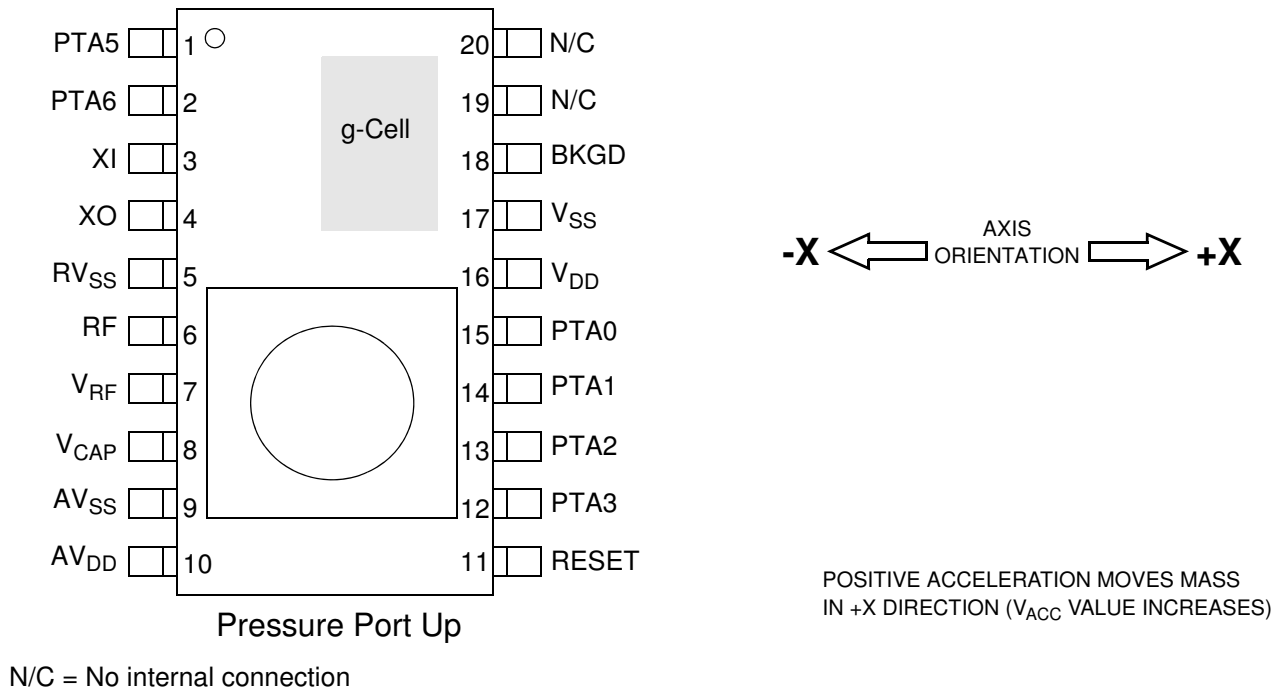

N/C = No internal connection

Figure 2-1 MPXY8300 Series 20-Pin SOIC Package Pinout

Figure 2-2 Port Up Lead Forming

2.2 Recommended Application
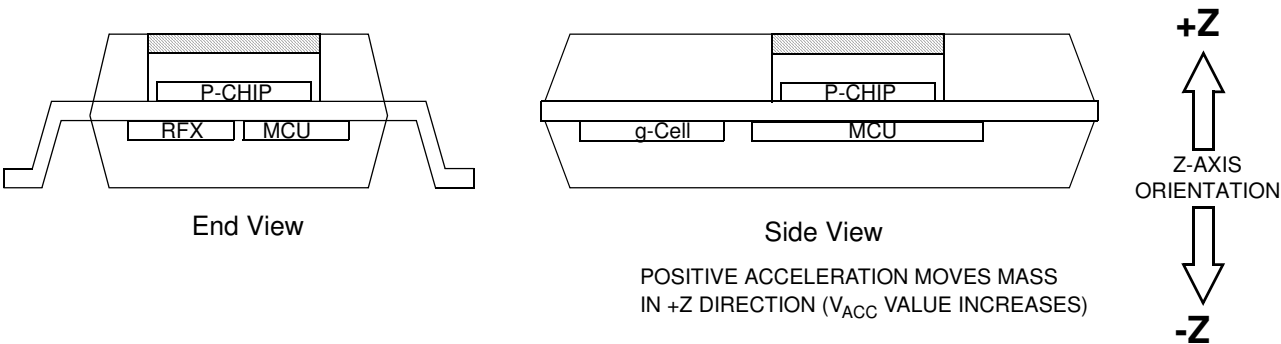
Examples of simple ASK/FSK tire pressure monitors using the internal PLL-based RF output stage is shown in Figure 2-3 and Figure 2-4. Any of the PTA0:3 and PTA5:6 pins can also be used as general purpose I/O pins. Any of the PTA0:3 or PTA5:6 pins that are not used in the application should be connected to V_{SS}.
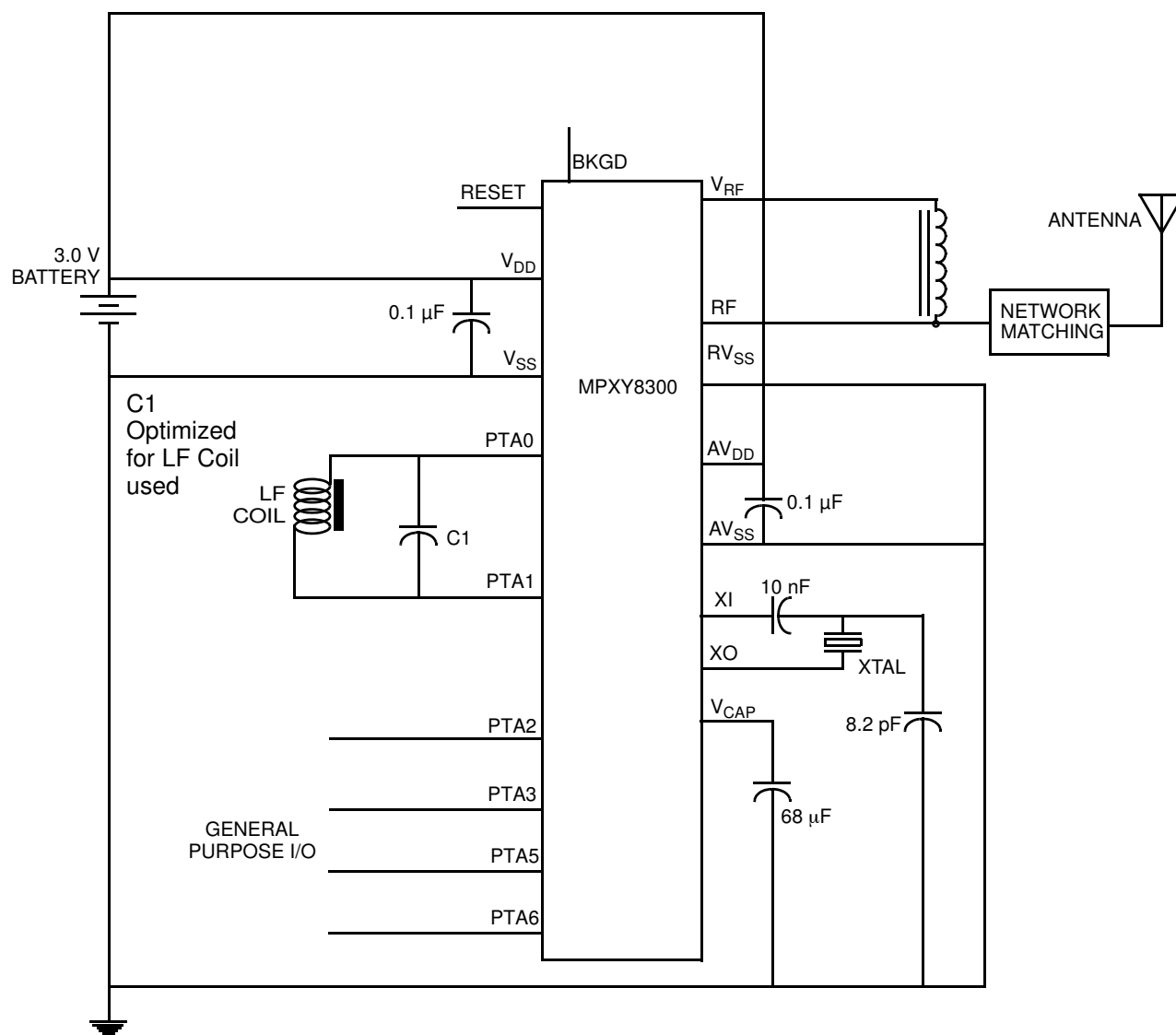

Figure 2-3 MPXY8300 Series Example Application

Figure 2-4 MPXY8300 Series Example Application (Using RFX Charge Pump)

2.3   Signal Properties

The following sections give a description of the general function of each pin.

2.3.1 V_{DD} and V_{SS} Pins

The digital circuits operate from a single power supply connected to the device through the V_{DD} and V_{SS} pins. V_{DD} is the positive supply and V_{SS} is the ground. The conductors to the power supply should be connected to the V_{DD} and V_{SS} pins and locally decoupled as shown in Figure 2-5.

Care should be taken to reduce measurement signal noise by separating the V_{DD}, V_{SS}, AV_{DD}, AV_{SS} and RV_{SS} pins using a "star" connection such that each metal trace does not share any load currents with other external devices as shown in Figure 2-5.
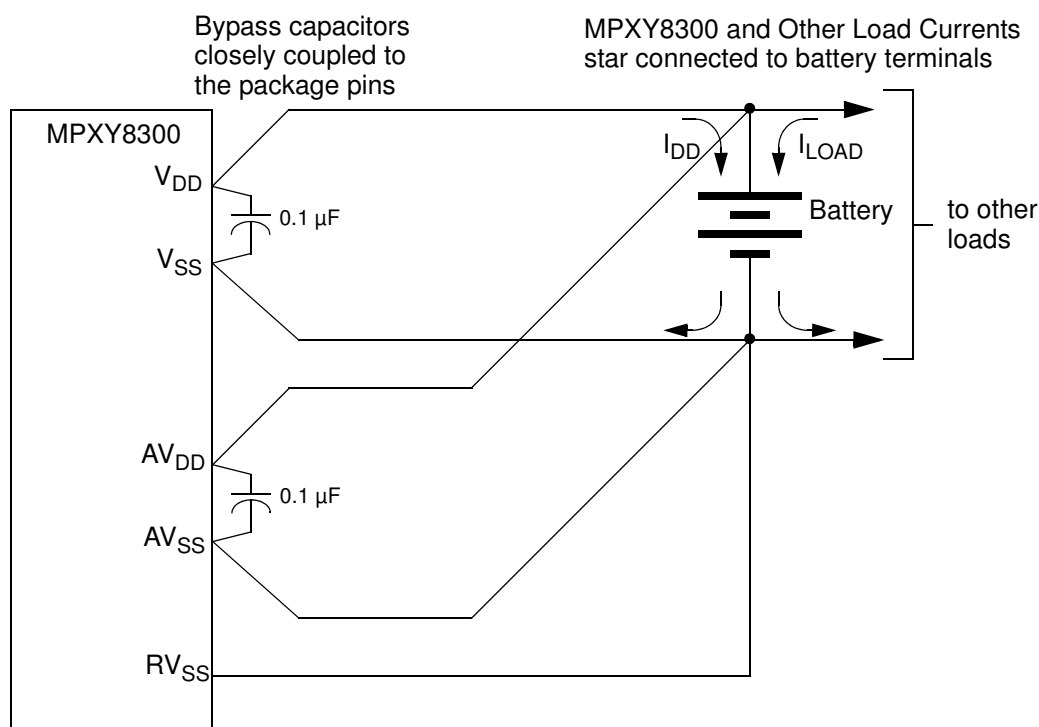
Figure 2-5 Recommended Power Supply Connections

2.3.2   AV_DD and AV_SS Pins

The analog circuits operate from a single power supply connected to the device through the AV_{DD} and AV_{SS} pins. AV_{DD} is the positive supply and AV_{SS} is the ground. The conductors to the power supply should be connected to the AV_{DD} and AV_{SS} pins and locally decoupled as shown in Figure 2-5.

Care should be taken to reduce measurement signal noise by separating the V_{DD}, V_{SS}, AV_{DD}, AV_{SS} and RV_{SS} pins using a "star" connection such that each metal trace does not share any load currents with other external devices as shown in Figure 2-5.

2.3.3   V_CAP Pin

An external load capacitor is connected the V_{CAP} pin when using the internal charge pump to generate a voltage higher than V_{DD} for the RFX device. If the charge pump is not used this pin should be left unconnected.

2.3.4   V_RF Pin

The V_{RF} pin provides the power output voltage for RFX and should be connected to the RF pin through an inductor.

2.3.5   RV_SS Pin

Power in the RF output amplifier is returned to the supply through the RV_{SS} pin. This conductor should be connected to the power supply as shown in Figure 2-5 using a "star" connection such that each metal trace does not share any load currents with other supply pins.

2.3.6   RF Pin

The RF pin is the RF energy data supplied by the device to an external antenna.

2.3.7   XO, XI Pins

The XO and XI pins are for an external crystal to be used by the internal PLL for creating the carrier frequencies and data rates for the RF pin.

MPXY8300 Series

2.3.8   PTA0:1 Pins

The PTA0:1 pins can be used by the LF receiver (LFR) as a differential input channel for sensing low level signals from an external low frequency (LF) coil. The external LF coil should be connected between the PTA0 and the PTA1 pins. Signaling into the LFR pins can place the device into various diagnostic or operational modes. The LFR is comprised of the detector and the decoder.

These two pins can also be configured as normal bi-directional I/O pins with a programmable pull-up device, but each will always have a pull-down resistor connected to V_{SS} of approximately 500 kohm.

2.3.9   PTA2:3 Pins

The PTA2:3 pins are general purpose I/O pins. These two pins can be configured as normal bi-directional I/O pins with programmable pull-up or pull-down devices and/or wake-up interrupt capability; or they can be connected to the two input channels of the Timer Pulse Width (TPM1) module.

2.3.10   PTA5:6 Pins

The PTA5:6 pins are general purpose I/O pins. These two pins can also be configured with programmable pull-up or pull-down devices and/or wake-up interrupt capability.

2.3.11   BKGD Pin

The BKGD pin is used to place the device in the background debug mode (BDM) to evaluate MCU code and to also transfer data to/from the internal memories as described in Section 3.3. If the BKGD pin is held low when the device comes out of reset the device will go into the active background debug mode (BDM). The active background debug mode can also be activated by specific toggling of the BKGD pin while the device is powered up if the MCU is not in a stop mode.

The BKGD pin has an internal pull-up device, but should be connected to V_{DD} in the application unless there is a need to enter BDM operation after the device as been soldered into a PWB. If in-circuit BDM is desired the BKGD pin should be connected to V_{DD} through a low impedance resistor (<10 kΩ) which can be overdriven by an external signal. This low impedance resistor reduces the possibility of getting into the debug mode in the application due to an EMC event.

2.3.12   RESET Pin

The RESET pin is used for test and establishing the BDM condition.

This pin has an internal pull-up device, but should be connected to V_{DD} in the application unless there is a need to enter BDM operation after the device as been soldered to the PWB. If in-circuit BDM is desired the RESET pin should be connected to V_{DD} through a low impedance resistor (<10 kΩ) which can be overdriven by an external signal. This low impedance resistor reduces the possibility of getting into the debug mode in the application due to an EMC event.

SECTION 3   MODES OF OPERATION

The operating modes of the MPXY8300 Series are described in this section. Entry into each mode, exit from each mode, and functionality while in each of the modes are described.

3.1   Features

- Active background debug mode for code development
- Wait mode:
    - CPU shuts down to conserve power
    - System clocks continue running
    - Full voltage regulation is maintained
- Stop modes:
    - System clocks stopped
    - Stop1 — Power down of most internal circuits, including RAM, for maximum power savings; voltage regulator in standby
    - Stop2 — Power down of most internal circuits; voltage regulator in standby; RAM maintained in loose regulation
    - Stop3 — All internal circuits powered; voltage regulator in standby with loose regulation
    - Stop4 — All internal circuits powered and full voltage regulation maintained for fastest recovery

3.2   Run Mode

This is the normal operating mode for the device. This mode is selected when the BKGD pin is high at the rising edge of reset. In this mode, the CPU executes code from internal memory following a reset with execution beginning at address $C000.

3.3   Active Background Mode

The active background mode functions are managed through the background debug controller (BDC) in the HCS08 core. The BDC, together with the on-chip debug module (DBG), provide the means for analyzing MCU operation during software development.

Active background mode is entered in any of five ways:
- When the BKGD pin is low at the rising edge of a power up reset
- When a BACKGROUND command is received through the BKGD pin
- When a BGND instruction is executed by the CPU
- When encountering a BDC breakpoint
- When encountering a DBG breakpoint

Once in active background mode, the CPU is held in a suspended state waiting for serial background commands rather than executing instructions from the user's application program. Background commands are of two types:
- Non-intrusive commands, defined as commands that can be issued while the user program is running. Non-intrusive commands can be issued through the BKGD pin while the MCU is in run mode; non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
    - Memory access commands
    - Memory-access-with-status commands
    - BDC register access commands
    - The BACKGROUND command
- Active background commands, which can only be executed while the MCU is in active background mode. Active background commands include commands to:
    - Read or write CPU registers
    - Trace one user program instruction at a time
    - Leave active background mode to return to the user's application program (GO)

The active background mode is used to program a bootloader or user application program into the FLASH program memory before the MCU is operated in run mode for the first time. When the device is shipped from the Freescale factory, the FLASH program memory is erased by default (unless specifically requested otherwise) so there is no program that could be executed in run mode until the FLASH memory is initially programmed.

The active background mode can also be used to erase and reprogram the FLASH memory after it has been previously programmed.

MPXY8300 Series

3.4 Wait Mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The I bit in CCR is cleared when the CPU enters the wait mode, enabling interrupts. When an interrupt request occurs, the CPU exits the wait mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

While the MCU is in wait mode, there are some restrictions on which background debug commands can be used. Only the BACKGROUND command and memory-access-with-status commands are available when the MCU is in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from wait mode and enter active background mode.

3.5 Stop Modes

One of four stop modes is entered upon execution of a STOP instruction when the STOPE bit in the system option register is set. In all stop modes, all internal clocks are halted except for the low frequency 1 kHz oscillator (LFO) which runs continuously whenever power is applied to the V_{DD} and V_{SS} pins. If the STOPE bit is not set when the CPU executes a STOP instruction, the MCU will not enter any of the stop modes and an illegal opcode reset is forced. The stop modes are selected by setting the appropriate bits in SPMSC2. Table 3-1 summarizes the behavior of the MCU in each of the stop modes.

3.5.1 Stop1 Mode

The stop1 mode provides the lowest possible standby power consumption by causing the internal circuitry of the MCU to be powered down.

When the MCU is in stop1 mode, all internal circuits that are powered from the voltage regulator are turned off. The voltage regulator is in a low-power standby state. Exit from stop1 is done by asserting either a reset or an interrupt function to the MCU.

Entering stop1 mode automatically asserts LVD. Stop1 cannot be exited until $V_{DD} > V_{LVDH/L}$ rising (V_{DD} must rise above the LVI re-arm voltage).

Upon wake-up from stop1 mode, the MCU will start up as from a power-on reset (POR) by taking the reset vector.

NOTE

Specific to the tire pressure monitoring application the parameter registers and the LFO with wake-up timer are powered up at all times whenever voltage is applied to the supply pins. The LFR detector and MFO may be periodically powered up by the LFR decoder.

Table 3-1 Stop Mode Behavior

Mode	Stop1	Stop2	Stop3	Stop4
LFO Oscillator, PWU	Always On & Clocking			
MFO Oscillator[1]	Optionally On	Optionally On	Optionally On	Optionally On
HFO Oscillator	Off	Off	Off	Off
CPU	Off	Off	Standby	Standby
RAM	Off	Standby	Standby	Standby
Parameter Registers	On	On	On	On
FLASH	Off	Off	Standby	Standby
TPM1 2 Chan TImer/PWM	Off	Off	Off	Off
Digital I/O	Off	Off	Standby	Standby
P-Chip Interface (PCI)	Off	Off	Standby	Optionally On
P-Chip	Off	Off	Standby	Optionally On
Acceleration Interface (ACI)	Off	Off	Standby	Optionally On
Acceleration g-cells	Off	Off	Standby	Optionally On
Temperature Sensor (in ADC10)	Off	Off	Off	Optionally On[4]
Normal Temperature Restart	Optionally On	Optionally On	Optionally On	Optionally On
Voltage Reference (in ADC10)	Off	Off	Optionally On[4]	Optionally On[4]

Table 3-1 Stop Mode Behavior (Continued)

Mode	Stop1	Stop2	Stop3	Stop4
LFR Detector[2]	Periodically On	Periodically On	Periodically On	Periodically On
LFR Decoder	On	On	On	On
RFX Charge Pump	Optionally On	Optionally On	Optionally On	Optionally On
RFX Data Buffer, Encoder	On	On	On	On
RFX Transmitter[3]	Optionally On	Optionally On	Optionally On	Optionally On
ADC10	Off	Off	Standby	Optionally On[4]
SPI	Off	Off	Standby	Standby
Regulator	Off	Standby	Standby	On
I/O Pins	Hi-Z	States Held	States Held	States Held
Wake-Up Methods	Interrupts, resets	Interrupts, resets	Interrupts, resets	Interrupts, resets

NOTES:
1. MFO oscillator started if the LFR detectors are periodically sampled, the LFR detectors detect an input signal; or a pressure or acceleration reading is in progress.
2. Period of sampling set by MCU.
3. RF data buffer may be set up to run while the CPU is in the stop modes.
4. Requires internal ADC10 clock to be enabled.

3.5.2   Stop2 Mode

The stop2 mode provides very low standby power consumption and maintains the contents of RAM and the current state of all of the I/O pins.

Before entering stop2 mode, the user must save the contents of the I/O port registers, as well as any other memory-mapped registers which they want to restore after exit of stop2, to locations in RAM. Upon exit of stop2, these values can be restored by user software before the I/O pin latches are opened.

When the MCU is in stop2 mode, all internal circuits that are powered from the voltage regulator are turned off, except for the RAM. The voltage regulator is placed in a low-power standby state, as is the ADC10. Upon entry into stop2, the states of the I/O pins are latched.

Exit from stop2 is done by asserting either a reset or interrupt function to the MCU.

Upon wake-up from stop2 mode, the MCU will start up the same as from a power-on reset (POR) except the I/O pin states remain latched by taking the reset vector. The system and all peripherals will be in their default reset states and must be initialized. After waking up from stop2, the PPDF bit in SPMSC2 is set. This flag may be used to direct user code to go to a stop2 recovery routine. PPDF remains set and the I/O pin states remain latched until a logic 1 is written to PPDACK in SPMSC2.

To maintain I/O state for pins that were configured as general-purpose I/O, the user must restore the contents of the I/O port registers, which have been saved in RAM, to the port registers before writing to the PPDACK bit. If the port registers are not restored from RAM before writing to PPDACK, then the register bits will assume their default reset states when the I/O pin latches are opened and the I/O pins will switch to their reset states.

For pins that were configured as peripheral I/O, the user must reconfigure the peripheral module that interfaces to the pin before writing to the PPDACK bit. If the peripheral module is not enabled before writing to PPDACK, the pins will be controlled by their associated port control registers when the I/O latches are opened.

3.5.3   Stop3 Mode

Upon entering the stop3 mode, all of the clocks in the MCU except the LFO are halted. The voltage regulator and the ADC10 enter into their standby states. The states of all of the internal registers and logic, as well as the RAM content, are maintained. The I/O pin states are not latched at the pin as in stop2. Instead they are maintained by virtue of the fact that the states of the internal logic driving the pins are being maintained.

Exit from stop3 is done by asserting either a reset or interrupt function to the MCU. If caused by a reset the MCU will be reset and operation will resume after taking the reset vector. Exit by means of an asynchronous interrupt or the real-time interrupt will result in the MCU taking the appropriate interrupt vector.

3.5.4   Stop4 Mode

Stop4 is identical to stop3 except that full regulation is maintained which results in higher power consumption.

3.5.5   LVD Enabled in Stop Mode

The LVD system is capable of generating either an interrupt or a reset when the supply voltage drops below the LVD voltage. If the LVD is enabled by setting the LVDE and the LVDSE bits in SPMSC1 when the CPU executes a STOP instruction, then the voltage regulator remains active during stop mode. If the user attempts to enter either stop1 or stop2 with the LVD enabled in stop (LVDSE = 1), the MCU will enter stop4 instead.

3.5.6   Active BDM Enabled in Stop Mode

Entry into the active background debug mode from run mode is enabled if the ENBDM bit in BDCSCR is set. The BDCSCR register is not memory mapped so it can only be accessed through the BDM interface by use of the BDM commands READ_STATUS and WRITE_CONTROL. If ENBDM is set when the CPU executes a STOP instruction, the system clocks to the background debug logic remain active when the MCU enters stop mode so background debug communication is still possible. In addition, the voltage regulator does not enter its low-power standby state but maintains full internal regulation (stop4 mode). If the user attempts to enter either stop1 or stop2 with ENDBM set, the MCU will enter stop4 with system clocks running.

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the ENDBM bit is set. Once in background debug mode, all background commands are available. The table below summarizes the behavior of the MCU in stop mode when entry into the background debug mode is enabled.

3.5.7   MCU On-Chip Peripheral Modules in Stop Modes

When the MCU enters any stop mode, system clocks to the internal peripheral modules except the wake-up timer and LFR detectors/decoder are stopped. Even in the exception case (ENDBM = 1), where clocks are kept alive to the background debug logic, clocks to the peripheral systems are halted to reduce power consumption.

I/O Pins (Optional PTA0:3)

The I/O pin states remain unchanged when the MCU enters stop3 or stop4 mode.

If the MCU is configured to go into stop2 mode, the I/O pin states are latched before entering stop2.

If the MCU is configured to go into stop1 mode, the I/O pins are forced to their default reset state (Hi-Z) upon entry into stop1.

Memory

All RAM and register contents are preserved while the MCU is in stop3 or stop4 mode.

All registers will be reset upon wake-up from stop2, but the contents of RAM are preserved and pin states remain latched until the PPDACK bit is written. The user may save any memory-mapped register data into RAM before entering stop2 and restore the data upon exit from stop2.

All registers will be reset upon wake-up from stop1 and the contents of RAM are not preserved. The MCU must be initialized as upon reset. The contents of the FLASH memory are non-volatile and are preserved in any of the stop modes.

Parameter Registers

The 32 bytes of parameter registers are kept active in all modes of operation as long as power is applied to the supply pins. The contents of the parameter registers behave like RAM and are unaffected by any reset.

LFO

The LFO remains active regardless of any mode of operation.

MFO

The medium frequency oscillator (MFO) will remain powered up when the MCU enters the stop mode if the LFR detector is periodically sampled or a pressure or acceleration measurement has been initiated.

CFO

The CFO can be activated in any of the stop modes, but will shut itself off after the selected T_{CHG} delay.

HFO

The HFO is halted in all stop modes.

PWU

The PWU remains active regardless of any mode of operation.

ADC10

If the asynchronous internal ADC10 clock is not selected as the conversion clock, entering the stop3 or stop4 mode aborts the existing conversion and places the ADC10 in its idle state. After exiting from stop mode, a write to the ADSC1 register is required to resume conversions.

If the internal asynchronous ADC10 clock is selected as the conversion clock, the ADC10 can continue operation during stop4 mode. If a conversion is in progress when the MCU enters stop4 mode, it continues until completion. Conversions can be initiated while the MCU is in stop4 mode by means of the hardware trigger or if continuous conversions are enabled.

Either a conversion complete or compare interrupt will wake the MCU from stop4 mode. The ADC10 can operate as described above with the MCU in stop3 mode but the accuracy of the results are not guaranteed.

The ADC10 module is automatically disabled when the MCU enters either stop1 or stop2 mode. All ADC10 module registers contain their reset values following exit from stop1 or stop2. Therefore the ADC10 module must be re-enabled and re-configured following exit from stop1 or stop2.

LFR

When the MCU enters stop mode the detectors in the LFR will remain powered up depending on the states of the LFE0:1 and LFM0:1 bits and the periodic sampling. Refer to Section 12 for more details.

Bandgap Reference

The bandgap reference is disabled in all stop modes except stop4.

TPM1

When the MCU enters stop mode, the clock to the TPM1 module stops and the module halts operation. If the MCU is configured to go into stop2 or stop1 mode, the TPM1 module will be reset upon wake-up from stop and must be re initialized.

SPI Master

When the MCU enters stop mode, the clocks to the SPI module stop; the module halts operation; and communication with the RFX is ended. If the MCU is configured to go into stop2 or stop1 mode, the SPI module will be reset upon wake-up from stop and must be re-initialized.

Voltage Regulator

The voltage regulator enters a low-power standby state when the MCU enters any of the stop modes except stop4 (LVDSE = 1 or ENBDM =1).

Temperature Sensor

The temperature sensor is disabled in all stop modes except stop4.

Temperature Restart

When the MCU enters a stop mode the temperature restart will remain powered up if the TRE bit is set. If the temperature restart level is reached the MCU will restart from the reset vector.

3.5.8 RFX On-Chip Peripheral Modules in Stop Modes

When the MCU enters any stop mode the modules on the RFX may or may not power down depending on their respective control bits.

RF Output

When the MCU enters stop mode the external crystal oscillator (XCO), bit rate generator, PLL, VCO, RF data buffer, data encoder, and RF output stage will remain powered up if the SEND bit is set.

SPI Slave

When the MCU enters stop mode, the SPI slave module remains powered up but halts operation because the SPI master on the MCU is inactive.

Charge Pump

When the MCU enters stop mode, the RFX charge pump will remain powered up if the CPUMP bit has been set and the charge time has not elapsed and the VCAP voltage has not reached the maximum threshold, V_{CAPMAX}.

MPXY8300 Series

SECTION 4   MEMORY

The overall memory map of the MPXY8300 Series resides mainly in the MCU. There are additional memory elements in the P-Chip which are accessed by the PCI as described in Section 10.3. There are also additional memory elements in the RFX which are accessed by the internal SPI as described in Figure 13.These accesses to the P-Chip and RFX are performed using the firmware routines provided by Freescale in the FLASH firmware area as described in Section 14 and are not normally accessed by the user.

4.1   MCU Memory Map

As shown in Figure 4-1, MCU on-chip memory in the MPXY8300 Series consists of parameter registers, RAM, FLASH program memory for nonvolatile data storage, and I/O and control/status registers.

DIRECT PAGE REGISTERS	$0000 — $003F
PARAMETER REGISTERS	$0040 — $005F
RAM 512 BYTES	$0060 — $025F
UNIMPLEMENTED 5536 BYTES	$0260 — $17FF
HIGH PAGE REGISTERS	$1800 — $182B
UNIMPLEMENTED 42964 BYTES	$182C — $BFFF
USER FLASH 8160 BYTES	$C000 — $DFDF
USER VECTORS	$DFE0 — $DFFF
FIRMWARE JUMP TABLE	$E000 — $E03F
FIRMWARE FLASH 8128 BYTES	$E040 — $FFFF

Figure 4-1 MPXY8300 Series MCU Memory Map

The total memory map is 16K, but the upper 8K is used for firmware and test software. Upon power up the firmware will initialize the device and redirect all vectors to the user area from $DFC0 through $DFFF. Any calls to the firmware subroutines are accessed through the jump table located at $E000 through $E03F.

4.2   Reset and Interrupt Vectors

Table 4-1 shows address assignments for reset and interrupt vectors. The vector names shown in this table are the labels used in the equate file provided by Freescale in the Codewarrior project file.

Table 4-1 Vector Summary

Vector Addr (High/Low)	Module Source	Vector Name
$DFE0:DFE1	KBI	Vkbi
$DFE2:DFE3	PCI	Vpci
$DFE4:DFE5	ACI	Vacc
$DFE6:DFE7	Sys Ctrl	Vrti
$DFE8:DFE9	LFR	Vlfrcvr
$DFEA:DFEB	ADC10	Vadc1
$DFEC:DFED	Reserved	
$DFEE:DFEF	SPI	Vspi1
$DFF0:DFF1	TPM1	Vtpm1ovf
$DFF2:DFF3	TPM1	Vtpm1ch1
$DFF4:DFF5	TPM1	Vtpm1ch0
$DFF6:DFF7	PWU	Vwuktmr
$DFF8:DFF9	Sys Ctrl	Vlvd
$DFFA:DFFB	Reserved	
$DFFC:DFFD	SWI opcode	Vswi
$DFFE:DFFF	Sys Ctrl - POR	Vreset
	Sys Ctrl - PRF	
	Sys Ctrl - COP	
	Sys Ctrl - LVD	
	Temp Restart	
	Illegal opcode	
	Illegal address	

4.3   Register Addresses and Bit Assignments

The registers in the MPXY8300 Series are divided into these four groups:
   • Direct-page registers are located in the first 64 locations in the memory map; these are accessible with efficient direct addressing mode instructions.
   • The parameter registers begin at address $0040; these are also accessible with efficient direct addressing mode instructions.
   • High-page registers are used less often, so they are located above $1800 in the memory map. This leaves more room in the direct page for more frequently used registers and variables.
   • The nonvolatile register area consists of a block of 16 locations in FLASH memory at $FFB0:FFBF. Nonvolatile register locations include:
       – Three values that are loaded into working registers at reset
       – An 8-byte backdoor comparison key that optionally allows the user to gain controlled access to secure memory.
   Because the nonvolatile register locations are FLASH memory, they must be erased and programmed like other FLASH memory locations.

Direct page registers are located within the first 256 locations in the memory map, so they are accessible with efficient direct addressing mode instructions, which requires only the lower byte of the address. Bit manipulation instructions can be used to access any bit in any direct-page register. Table 4-2 is a summary of all user-accessible direct-page registers and control bits. Those related to the TPMS application and modules are described in detail in this specification.

The register names in the second column of the table are shown in bold to set them apart from the bit names to the right. Cells that are not associated with named bits are shaded. A shaded cell with a 0 indicates this unused bit always reads as a 0. Shaded cells with dashes indicate unused or reserved bit locations that could read as 1s or 0s.

MPXY8300 Series

Table 4-2 MCU Direct Page Register Summary

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x0000	PTAD	—	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
0x0001	PTAPE	—	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
0x0002	Reserved	—	—	—	—	—	—	—	—
0x0003	PTADD	—	PTADD6	PTADD5	PTADD4	PTADD3	PTADD2	PTADD1	PTADD0
0x0004	PTBD	—	PTBD6	PTBD5	PTBD4	PTBD3	PTBD2	PTBD1	PTBD0
0x0005	PTBPE	—	PTBPE6	PTBPE5	PTBPE4	PTBPE3	PTBPE2	PTBPE1	PTBPE0
0x0006	Reserved	—	—	—	—	—	—	—	—
0x0007	PTBDD	—	PTBDD6	PTBDD5	PTBDD4	PTBDD3	PTBDD2	PTBDD1	PTBDD0
0x0008	PTCD	—	—	—	—	—	PTCD2	PTCD1	PTCD0
0x0009	PTCPE	—	—	—	—	—	PTCPE2	PTCPE1	PTCPE0
0x000A	Reserved	—	—	—	—	—	—	—	—
0x000B	PTCDD	—	—	—	—	—	PTCDD2	PTCDD1	PTCDD0
0x000C	KBISC	0	0	0	0	KBF	KBACK	KBIE	KBIMOD
0x000D	KBIPE	0	0	0	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
0x000E	KBIES	0	0	0	KBEDG4	KBEDG3	KBEDG2	KBEDG1	KBEDG0
0x000F	IRQSC	0	0	IRQEDG	IRQPE	IRQF	IRQACK	IRQIE	IRQMOD
0x0010	TPMSC	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
0x0011	TPMCNTH	Bit 15	14	13	12	11	10	9	Bit 8
0x0012	TPMCNTL	Bit 7	6	5	4	3	2	1	Bit 0
0x0013	TPMMODH	Bit 15	14	13	12	11	10	9	Bit 8
0x0014	TPMMODL	Bit 7	6	5	4	3	2	1	Bit 0
0x0015	TPMC0SC	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	0	0
0x0016	TPMC0VH	Bit 15	14	13	12	11	10	9	Bit 8
0x0017	TPMC0VL	Bit 7	6	5	4	3	2	1	Bit 0
0x0018	TPMC1SC	CH1F	CH1IE	MS1B	MS1A	ELS1B	ELS1A	0	0
0x0019	TPMC1VH	Bit 15	14	13	12	11	10	9	Bit 8
0x001A	TPMC1VL	Bit 7	6	5	4	3	2	1	Bit 0
0x001B–0x001F	Reserved	—	—	—	—	—	—	—	—
0x0020	SPIC1	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
0x0021	SPIC2	0	0	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
0x0022	SPIBR	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
0x0023	SPIS	SPRF	0	SPTEF	MODF	0	0	0	0
0x0024	Reserved	0	0	0	0	0	0	0	0
0x0025	SPID	Bit 7	6	5	4	3	2	1	Bit 0
0x0026–0x0027	Reserved	—	—	—	—	—	—	—	—
0x0028	LFCA0	LFA7	LFA6	LFA5	LFA4	LFA3	LFA2	LFA1	LFA0
0x0029	LFCA1	LFA15	LFA14	LFA13	LFA12	LFA11	LFA10	LFA9	LFA8
0x002A	LFCB0	LFB7	LFB6	LFB5	LFB4	LFB3	LFB2	LFB1	LFB0
0x002B	LFCB1	LFB15	LFB14	LFB13	LFB12	LFB11	LFB10	LFB9	LFB8
0x002C	LFDATA	LFD7	LFD6	LFD5	LFD4	LFD3	LFD2	LFD1	LFD0
0x002D	LFCR	LFM1	LFM0	LFCT	LFON	LFS3	LFS2	LFS1	LFS0
0x002E	LFCS0	LFDO	LFIE	LFOFF	LFRST	LFEN	LFWU8	LFLVL	LFAGC
0x002F	LFCS1	LFDF	LFBPF	LFCF	0	LFBF	LFAF	MDDO	LFIG
0x0030	ADSC1	COCO	AIEN	ADCO	ADCH				

Table 4-2 MCU Direct Page Register Summary (Continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0x0031	ADSC2	ADACT	ADTRG	ACFE	ACFGT	0	0	—	—
0x0032	ADRH	0	0	0	0	ADR11	ADR10	ADR9	ADR8
0x0033	ADRL	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
0x0034	ADCVH	0	0	0	0	0	0	ADCV9	ADCV8
0x0035	ADCVL	ADCV7	ADCV6	ADCV5	ADCV4	ADCV3	ADCV2	ADCV1	ADCV0
0x0036	ADCFG	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
0x0037	APCTL1	ADPC7	ADPC6	ADPC5	ADPC4	ADPC3	ADPC2	ADPC1	ADPC0
0x0038	PWUDIV	CLK1	CLK0	WDIV					
0x0039	PWUCS0	WUF	WUFACK	WUT					
0x003A	PWUCS1	PRF	PRFACK	PRST					
0x003B–0x003F	Reserved	—	—	—	—	—	—	—	—
		—	—	—	—	—	—	—	—
0x0040–0x005F	PARAM0–PARAM31	PARAMETER REGISTERS							

High-page registers are used much less often, so they are located above $1800 in the memory map. The high address registers control system level features as given in Table 4-3.

Table 4-3 MCU High Address Register Summary

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$1800	SRS	POR	PIN	COP	ILOP	ILAD	PWU	LVD	0
$1801	SBDFR	0	0	0	0	0	0	0	BDFR
$1802	SOPT1	COPE	COPCLKS	STOPE	1	0	0	BKGDPE	1
$1803	SOPT2	TRE	COPT2	COPT1	COPT0	LFOSEL	TCLKDIV	BUSCLKS1	BUSCLKS0
$1804	Reserved	0	0	0	0	0	0	0	0
$1805	Reserved	—	—	—	—	—	—	—	—
$1806	SDIDH	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
$1807	SDIDL	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
$1808	SRTISC	RTIF	RTIACK	RTICLKS	RTIE	0	RTIS2	RTIS1	RTIS0
$1809	SPMSC1	LVDF	LVDACK	LVDIE	LVDRE	LVDSE	LVDE	0[1]	BGBE
$180A	SPMSC2	0	0	0	PDF	PPDF	PPDACK	PDC	PPDC
$180B	Reserved	—	—	—	—	—	—	—	—
$180C	SPMSC3	LVWF	LVWACK	LVDV	LVWV	—	—	—	—
$180D	SPCITRM1	T7	T6	T5	T4	T3	T2	T1	T0
$180E	SPCITRM2	0	0	0	0	0	T10	T9	T8
$180F	SIMTST	—	—	—	—	—	—	—	TRO
$1810	DBGCAH	Bit 15	14	13	12	11	10	9	Bit 8
$1811	DBGCAL	Bit 7	6	5	4	3	2	1	Bit 0
$1812	DBGCBH	Bit 15	14	13	12	11	10	9	Bit 8
$1813	DBGCBL	Bit 7	6	5	4	3	2	1	Bit 0
$1814	DBGFH	Bit 15	14	13	12	11	10	9	Bit 8
$1815	DBGFL	Bit 7	6	5	4	3	2	1	Bit 0
$1816	DBGC	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
$1817	DBGT	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
$1818	DBGS	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0

Table 4-3 MCU High Address Register Summary (Continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$1819 – $181F	Reserved	—	—	—	—	—	—	—	—
$1820	FCDIV	DIVLD	PRDIV8	DIV5	DIV4	DIV3	DIV2	DIV1	DIV0
$1821	FOPT	KEYEN	FNORED	0	0	0	0	SEC01	SEC00
$1822	Reserved	—	—	—	—	—	—	—	—
$1823	FCNFG	0	0	KEYACC	0	0	0	0	0
$1824	FPROT	FPS7	FPS6	FPS5	FPS4	FPS3	FPS2	FPS1	FPDIS
$1825	FSTAT	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
$1826	FCMD	FCMD7	FCMD6	FCMD5	FCMD4	FCMD3	FCMD2	FCMD1	FCMD0
$1827 – $182B	Reserved	—	—	—	—	—	—	—	—

NOTES:
1. Bit 1 in $1809 is a reserved bit which must always be written to 0.

4.4 MCU Parameter Registers

The 32 bytes of parameter registers are located at addresses $0040 through $005F. These registers are powered up at all times and may be used to store temporary or history data during the times when the MCU is in any of the stop modes. The register at location $005F is used for interrupt flags by the firmware provided by Freescale. The assignment of the bits in location $005F are described in the Codewarrior project files supplied by Freescale.

4.5 RAM

The MPXY8300 Series includes static RAM. The locations in RAM below 0x0100 can be accessed using the more efficient direct addressing mode, and any single bit in this area can be accessed with the bit manipulation instructions (BCLR, BSET, BRCLR, and BRSET). Locating the most frequently accessed program variables in this area of RAM is preferred.

The RAM retains data when the MCU is in low-power wait stop2, stop3 or stop4 modes. At power-on or after wake-up from stop1, the contents of RAM are uninitialized. RAM data is unaffected by any reset provided that the supply voltage does not drop below the minimum value for RAM retention (V_{RAM}).

For compatibility with older M68HC05 MCUs, the HCS08 code in the device resets the stack pointer to 0x00FF. It is usually best to re initialize the stack pointer to the top of the RAM so the direct page RAM can be used for frequently accessed RAM variables and bit-addressable program variables. Include the following 2-instruction sequence in your reset initialization routine (where RamLast is equated to the highest address of the RAM in the equate file in the Codewarrior project file provided by Freescale).

```
LDHX     #RamLast+1     ;point one past RAM
TXS      ;SP<-(H:X-1)
```

When security is enabled, the RAM is considered a secure memory resource and is not accessible through BDM or through code executing from non-secure memory. See Section 4.6 for a detailed description of the security feature.

None of the RAM locations are used directly by the firmware provided by Freescale. The firmware routines utilize RAM only through stack operations; and the user needs to be aware of stack depth required by each routine as described in the Codewarrior project files supplied by Freescale.

4.6 FLASH

The FLASH memory is intended primarily for program storage. In-circuit programming allows the operating program to be loaded into the FLASH memory after final assembly of the application product. It is possible to program the entire array through the single-wire background debug interface. Because no special voltages are needed for FLASH erase and programming operations, in-application programming is also possible through other software-controlled communication paths. For a more detailed discussion of in-circuit and in-application programming, refer to the *HCS08 Family Reference Manual, Volume I,* Freescale document order number HCS08RMV1/D.

4.6.1 Features

Features of the FLASH memory include:
- User Program FLASH Size — 8192 bytes (16 pages of 512 bytes each)
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 100,000 program/erase cycles at typical voltage and temperature
- Flexible block protection
- Security feature for FLASH and RAM
- Auto power-down for low-frequency read accesses

4.6.2 Program and Erase Times

Before any program or erase command can be accepted, the FLASH clock divider register (FCDIV) must be written to set the internal clock for the FLASH module to a frequency (f_{FCLK}) between 150 kHz and 200 kHz (see Section 4.8.1). This register can be written only once, so normally this write is performed during reset initialization. FCDIV cannot be written if the access error flag, FACCERR in FSTAT, is set. The user must ensure that FACCERR is not set before writing to the FCDIV register. One period of the resulting clock ($1/f_{FCLK}$) is used by the command processor to time program and erase pulses. An integer number of these timing pulses are used by the command processor to complete a program or erase command.

Table 4-4 shows program and erase times. The bus clock frequency and FCDIV determine the frequency of FCLK (f_{FCLK}). The time for one cycle of FCLK is $t_{FCLK} = 1/f_{FCLK}$. The times are shown as a number of cycles of FCLK and as an absolute time for the case where $t_{FCLK} = 5$ μs. Program and erase times shown include overhead for the command state machine and enabling and disabling of program and erase voltages.

Table 4-4 Program and Erase Times

Parameter	Cycles of FCLK	Time if FCLK = 200 kHz
Byte program	9	45 μs
Byte program (burst)	4	20 μs[1]
Page erase	4000	20 ms
Mass erase	20,000	100 ms

NOTES:
1. Excluding start/end overhead

4.6.3 Program and Erase Command Execution

The steps for executing any of the commands are listed below. The FCDIV register must be initialized and any error flags cleared before beginning command execution. The command execution steps are:

1. Write a data value to an address in the FLASH array. The address and data information from this write is latched into the FLASH interface. This write is a required first step in any command sequence. For erase and blank check commands, the value of the data is not important. For page erase commands, the address may be any address in the 512-byte page of FLASH to be erased. For mass erase and blank check commands, the address can be any address in the FLASH memory. Whole pages of 512 bytes are the smallest block of FLASH that may be erased.

NOTE

Do not program any byte in the FLASH more than once after a successful erase operation. Reprogramming bits to a byte which is already programmed is not allowed without first erasing the page in which the byte resides or mass erasing the entire FLASH memory. Programming without first erasing may disturb data stored in the FLASH.

2. Write the command code for the desired command to FCMD. The five valid commands are blank check (0x05), byte program (0x20), burst program (0x25), page erase (0x40), and mass erase (0x41). The command code is latched into the command buffer.

3. Write a 1 to the FCBEF bit in FSTAT to clear FCBEF and launch the command (including its address and data information).

A partial command sequence can be aborted manually by writing a 0 to FCBEF any time after the write to the memory array and before writing the 1 that clears FCBEF and launches the complete command. Aborting a command in this way sets the FACCERR access error flag which must be cleared before starting a new command.

A strictly monitored procedure must be obeyed or the command will not be accepted. This minimizes the possibility of any unintended changes to the FLASH memory contents. The command complete flag (FCCF) indicates when a command is complete. The command sequence must be completed by clearing FCBEF to launch the command. Figure 4-2 is a flowchart for executing all of the commands except for burst programming. The FCDIV register must be initialized before using any FLASH commands. This must be done only once following a reset.
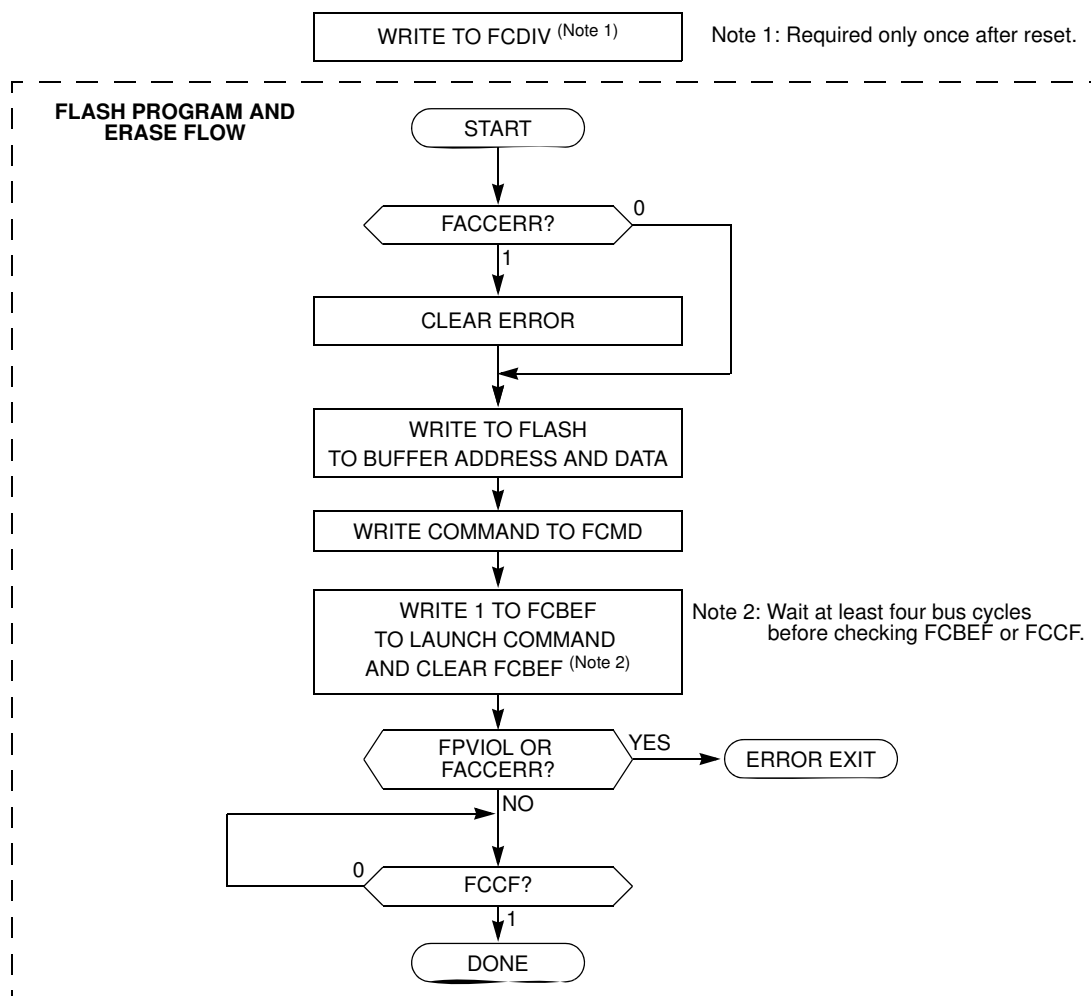

Figure 4-2 FLASH Program and Erase Flowchart

4.6.4   Burst Program Execution

The burst program command is used to program sequential bytes of data in less time than would be required using the standard program command. This is possible because the high voltage to the FLASH array does not need to be disabled between program operations. Ordinarily, when a program or erase command is issued, an internal charge pump associated with the FLASH memory must be enabled to supply high voltage to the array. Upon completion of the command, the charge pump is turned off. When a burst program command is issued, the charge pump is enabled and then remains enabled after completion of the burst program operation if these two conditions are met:

- The next burst program command has been queued before the current program operation has completed.
- The next sequential address selects a byte on the same physical row as the current byte being programmed. A row of FLASH memory consists of 64 bytes. A byte within a row is selected by addresses A5 through A0. A new row begins when addresses A5 through A0 are all zero.

The first byte of a series of sequential bytes being programmed in burst mode will take the same amount of time to program as a byte programmed in standard mode. Subsequent bytes will program in the burst program time provided that the conditions above are met. In the case the next sequential address is the beginning of a new row, the program time for that byte will be the standard time instead of the burst time. This is because the high voltage to the array must be disabled and then enabled again. If a new burst command has not been queued before the current command completes, then the charge pump will be disabled and high voltage removed from the array.
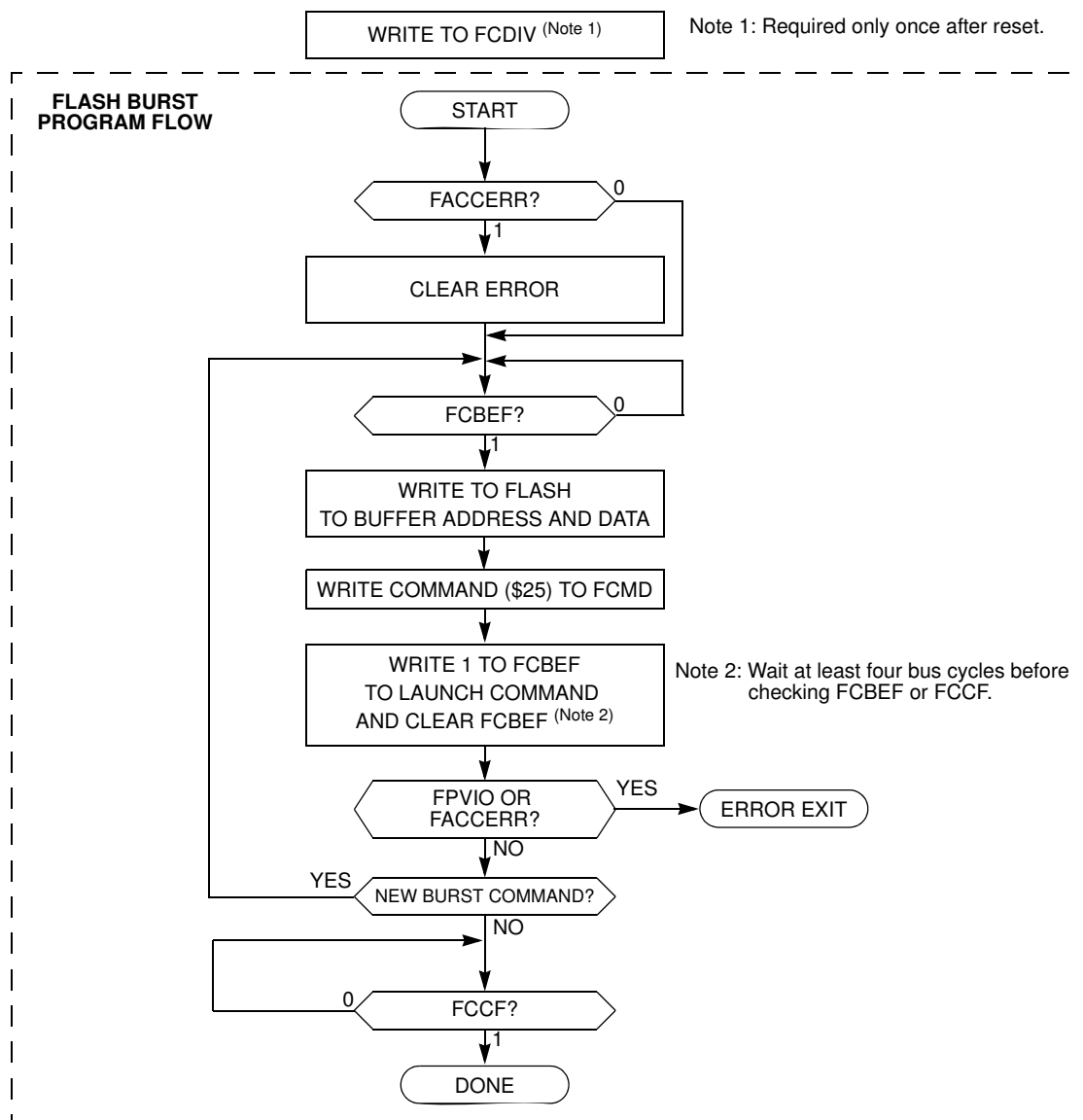
Figure 4-3 FLASH Burst Program Flowchart

4.6.5   Access Errors

An access error occurs whenever the command execution protocol is violated.

Any of the following specific actions will cause the access error flag (FACCERR) in FSTAT to be set. FACCERR must be cleared by writing a 1 to FACCERR in FSTAT   before any command can be processed.

- Writing to a FLASH address before the internal FLASH clock frequency has been set by writing to the FCDIV register
- Writing to a FLASH address while FCBEF is not set (A new command cannot be started until the command buffer is empty.)
- Writing a second time to a FLASH address before launching the previous command (There is only one write to FLASH for every command.)
- Writing a second time to FCMD before launching the previous command (There is only one write to FCMD for every command.)
- Writing to any FLASH control register other than FCMD after writing to a FLASH address
- Writing any command code other than the five allowed codes (0x05, 0x20, 0x25, 0x40, or 0x41) to FCMD
- Accessing (read or write) any FLASH control register other than the write to FSTAT (to clear FCBEF and launch the command) after writing the command to FCMD.
- The MCU enters stop mode while a program or erase command is in progress (The command is aborted.)

MPXY8300 Series

- Writing the byte program, burst program, or page erase command code (0x20, 0x25, or 0x40) with a background debug command while the MCU is secured (The background debug controller can only do blank check and mass erase commands when the MCU is secure.)
- Writing 0 to FCBEF to cancel a partial command

4.6.6 FLASH Block Protection

The block protection feature prevents the protected region of FLASH from program or erase changes. Block protection is controlled through the FLASH Protection Register (FPROT). When enabled, block protection begins at any 512 byte boundary below the last address of FLASH, 0xFFFF (see Section 4.8.4).

After exit from reset, FPROT is loaded with the contents of the NVPROT location which is in the nonvolatile register block of the FLASH memory. FPROT cannot be changed directly from application software so a runaway program cannot alter the block protection settings. Because NVPROT is within the last 512 bytes of FLASH, if any amount of memory is protected, NVPROT is itself protected and cannot be altered (intentionally or unintentionally) by the application software. FPROT can be written through background debug commands which allows a way to erase and reprogram a protected FLASH memory.

The block protection mechanism is illustrated below. The FPS bits are used as the upper bits of the last address of unprotected memory. This address is formed by concatenating FPS7:FPS1 with logic 1 bits as shown. For example, in order to protect the last 8192 bytes of memory (addresses 0xE000 through 0xFFFF), the FPS bits must be set to 1101 111 which results in the value 0xDFFF as the last address of unprotected memory. In addition to programming the FPS bits to the appropriate value, FPDIS (bit 0 of NVPROT) must be programmed to logic 0 to enable block protection. Therefore the value 0xDE must be programmed into NVPROT to protect addresses 0xE000 through 0xFFFF.
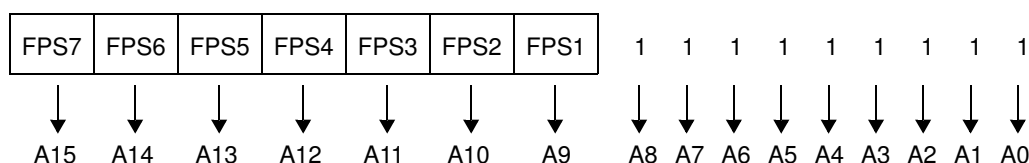
FPS7	FPS6	FPS5	FPS4	FPS3	FPS2	FPS1	1	1	1	1	1	1	1	1	1
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

Figure 4-4 Block Protection Mechanism

One use for block protection is to block protect an area of FLASH memory for a bootloader program. This bootloader program then can be used to erase the rest of the FLASH memory and reprogram it. Because the bootloader is protected, it remains intact even if MCU power is lost in the middle of an erase and reprogram operation.

4.6.7 Vector Redirection

Whenever any block protection is enabled, the reset and interrupt vectors will be protected. Vector redirection allows users to modify interrupt vector information without unprotecting bootloader and reset vector space. Vector redirection is enabled by programming the FNORED bit in the NVOPT register located at address 0xFFBF to zero. For redirection to occur, at least some portion but not all of the FLASH memory must be block protected by programming the NVPROT register located at address 0xFFBD. All of the interrupt vectors (memory locations 0xFFC0–0xFFFD) are redirected, though the reset vector (0xFFFE:FFFF) is not.

For example, if 512 bytes of FLASH are protected, the protected address region is from 0xFE00 through 0xFFFF. The interrupt vectors (0xFFC0–0xFFFD) are redirected to the locations 0xFDC0–0xFDFD. Now, if an SPI interrupt is taken for instance, the values in the locations 0xFDE0:FDE1 are used for the vector instead of the values in the locations 0xFFE0:FFE1. This allows the user to reprogram the unprotected portion of the FLASH with new program code including new interrupt vector values while leaving the protected area, which includes the default vector locations, unchanged.

4.7 Security

The MPXY8300 Series includes circuitry to prevent unauthorized access to the contents of FLASH and RAM memory. When security is engaged, FLASH and RAM are considered secure resources. Direct-page registers, high-page registers, and the background debug controller are considered unsecured resources. Programs executing within secure memory have normal access to any MCU memory locations and resources. Attempts to access a secure memory location with a program executing from an unsecured memory space or through the background debug interface are blocked (writes are ignored and reads return all 0s).

Security is engaged or disengaged based on the state of two nonvolatile register bits (SEC01:SEC00) in the FOPT register. During reset, the contents of the nonvolatile location NVOPT are copied from FLASH into the working FOPT register in high-page register space. A user engages security by programming the NVOPT location, which can be done at the same time the FLASH memory is programmed. The 1:0 state disengages security and the other three combinations engage security. Notice the erased state (1:1) makes the MCU secure. During development, whenever the FLASH is erased, it is good practice to immediately

program the SEC00 bit to 0 in NVOPT so SEC01:SEC00 = 1:0. This would allow the MCU to remain unsecured after a subsequent reset.

The on-chip debug module cannot be enabled while the MCU is secure. The separate background debug controller can still be used for background memory access commands, but the MCU cannot enter active background mode except by holding BKGD/MS low at the rising edge of reset.

A user can choose to allow or disallow a security unlocking mechanism through an 8-byte backdoor security key. If the nonvolatile KEYEN bit in NVOPT/FOPT is 0, the backdoor key is disabled and there is no way to disengage security without completely erasing all FLASH locations. If KEYEN is 1, a secure user program can temporarily disengage security by:

1. Writing 1 to KEYACC in the FCNFG register. This makes the FLASH module interpret writes to the backdoor comparison key locations (NVBACKKEY through NVBACKKEY+7) as values to be compared against the key rather than as the first step in a FLASH program or erase command.

2. Writing the user-entered key values to the NVBACKKEY through NVBACKKEY+7 locations. These writes must be done in order starting with the value for NVBACKKEY and ending with NVBACKKEY+7. STHX must not be used for these writes because these writes cannot be done on adjacent bus cycles. User software normally would get the key codes from outside the MCU system through a communication interface such as a serial I/O.

3. Writing 0 to KEYACC in the FCNFG register. If the 8-byte key that was just written matches the key stored in the FLASH locations, SEC01:SEC00 are automatically changed to 1:0 and security will be disengaged until the next reset.

The security key can be written only from secure memory (either RAM or FLASH), so it cannot be entered through background commands without the cooperation of a secure user program.

The backdoor comparison key (NVBACKKEY through NVBACKKEY+7) is located in FLASH memory locations in the nonvolatile register space so users can program these locations exactly as they would program any other FLASH memory location. The nonvolatile registers are in the same 512-byte block of FLASH as the reset and interrupt vectors, so block protecting that space also block protects the backdoor comparison key. Block protects cannot be changed from user application programs, so if the vector space is block protected, the backdoor security key mechanism cannot permanently change the block protect, security settings, or the backdoor key.

Security can always be disengaged through the background debug interface by taking these steps:

1. Disable any block protections by writing FPROT. FPROT can be written only with background debug commands, not from application software.

2. Mass erase FLASH if necessary.

3. Blank check FLASH. Provided FLASH is completely erased, security is disengaged until the next reset.

To avoid returning to secure mode after the next reset, program NVOPT so SEC01:SEC00 = 1:0.

4.8   FLASH Registers and Control Bits

The FLASH module has nine 8-bit registers in the high-page register space, three locations in the nonvolatile register space in FLASH memory which are copied into three corresponding high-page control registers at reset. There is also an 8-byte comparison key in FLASH memory. Refer to Table 4-3 and Table 4-4 for the absolute address assignments for all FLASH registers. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

4.8.1   FLASH Clock Divider Register (FCDIV)

Bit 7 of this register is a read-only status flag. Bits 6 through 0 can be read at any time but can be written only once. Before any erase or programming operations are possible, write to this register to set the frequency of the clock for the nonvolatile memory system within acceptable limits.

	7	6	5	4	3	2	1	0
R	DIVLD	PRDIV8	DIV5	DIV4	DIV3	DIV2	DIV1	DIV0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved
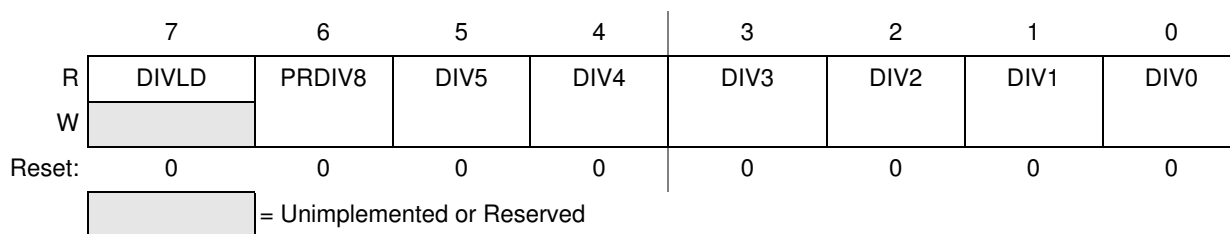
Figure 4-5 FLASH Clock Divider Register (FCDIV)

Table 4-5 FCDIV Register Field Descriptions

Field	Description
7 DIVLD	**Divisor Loaded Status Flag** — When set, this read-only status flag indicates that the FCDIV register has been written since reset. Reset clears this bit and the first write to this register causes this bit to become set regardless of the data written. 0 FCDIV has not been written since reset; erase and program operations disabled for FLASH 1 FCDIV has been written since reset; erase and program operations enabled for FLASH
6 PRDIV8	**Prescale (Divide) FLASH Clock by 8** 0 Clock input to the FLASH clock divider is the bus rate clock 1 Clock input to the FLASH clock divider is the bus rate clock divided by 8
5:0 DIV[5:0]	**Divisor for FLASH Clock Divider** — The FLASH clock divider divides the bus rate clock (or the bus rate clock divided by 8 if PRDIV8 = 1) by the value in the 6-bit DIV5:DIV0 field plus one. The resulting frequency of the internal FLASH clock must fall within the range of 200 kHz to 150 kHz for proper FLASH operations. Program/Erase timing pulses are one cycle of this internal FLASH clock which corresponds to a range of 5 μs to 6.7 μs. The automated programming logic uses an integer number of these pulses to complete an erase or program operation. • if PRDIV8 = 0 — $f_{FCLK} = f_{Bus} \div ([DIV5:DIV0] + 1)$ • if PRDIV8 = 1 — $f_{FCLK} = f_{Bus} \div (8 \times ([DIV5:DIV0] + 1))$ Table 4-6 shows the appropriate values for PRDIV8 and DIV5:DIV0 for selected bus frequencies.

Table 4-6 FLASH Clock Divider Settings

f_{Bus}	PRDIV8 (Binary)	DIV5:DIV0 (Decimal)	f_{FCLK}	Program/Erase Timing Pulse (5 μs Min, 6.7 μs Max)
20 MHz	1	12	192.3 kHz	5.2 μs
10 MHz	0	49	200 kHz	5 μs
8 MHz	0	39	200 kHz	5 μs
4 MHz	0	19	200 kHz	5 μs
2 MHz	0	9	200 kHz	5 μs
1 MHz	0	4	200 kHz	5 μs
200 kHz	0	0	200 kHz	5 μs
150 kHz	0	0	150 kHz	6.7 μs

4.8.2   FLASH Options Register (FOPT and NVOPT)

During reset, the contents of the nonvolatile location NVOPT are copied from FLASH into FOPT. Bits 5 through 2 are not used and always read 0. This register may be read at any time, but writes have no meaning or effect. To change the value in this register, erase and reprogram the NVOPT location in FLASH memory as usual and then issue a new MCU reset.

	7	6	5	4	3	2	1	0
R	KEYEN	FNORED	0	0	0	0	SEC01	SEC00
W								

Reset: This register is loaded from nonvolatile location NVOPT during reset.

[   ] = Unimplemented or Reserved

Figure 4-6 FLASH Options Register (FOPT)

Table 4-7 FOPT Register Field Descriptions

Field	Description
7 KEYEN	**Backdoor Key Mechanism Enable** — When this bit is 0, the backdoor key mechanism cannot be used to disengage security. The backdoor key mechanism is accessible only from user (secured) firmware. BDM commands cannot be used to write key comparison values that would unlock the backdoor key. For more detailed information about the backdoor key mechanism, refer to Section 4.7." 0　　No backdoor key access allowed 1　　If user firmware writes an 8-byte value that matches the nonvolatile backdoor key (NVBACKKEY through NVBACKKEY+7 in that order), security is temporarily disengaged until the next MCU reset
6 FNORED	**Vector Redirection Disable** — When this bit is 1, then vector redirection is disabled. 0　　Vector redirection enabled 1　　Vector redirection disabled
1:0 SEC0[1:0]	**Security State Code** — This 2-bit field determines the security state of the MCU as shown in Table 4-8. When the MCU is secure, the contents of RAM and FLASH memory cannot be accessed by instructions from any unsecured source including the background debug interface. For more detailed information about security, refer to Section 4.7. SEC01:SEC00 changes to 1:0 after successful backdoor key entry or a successful blank check of FLASH.

Table 4-8 Security States

SEC01:SEC00	Description
0:0	secure
0:1	secure
1:0	unsecured
1:1	secure

4.8.3 FLASH Configuration Register (FCNFG)

Bits 7 through 5 can be read or written at any time. Bits 4 through 0 always read 0 and cannot be written.

	7	6	5	4	3	2	1	0
R	0	0	KEYACC	0	0	0	0	0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-7 FLASH Configuration Register (FCNFG)

Table 4-9 FCNFG Register Field Descriptions

Field	Description
5 KEYACC	**Enable Writing of Access Key** — This bit enables writing of the backdoor comparison key. For more detailed information about the backdoor key mechanism, refer to Section 4.7. 0　　Writes to 0xFFB0–0xFFB7 are interpreted as the start of a FLASH programming or erase command 1　　Writes to NVBACKKEY (0xFFB0–0xFFB7) are interpreted as comparison key writes

4.8.4 FLASH Protection Register (FPROT and NVPROT)

During reset, the contents of the nonvolatile location NVPROT is copied from FLASH into FPROT. Bits 0, 1, and 2 are not used and each always reads as 0. This register can be read at any time, but user program writes have no meaning or effect. Background debug commands can write to FPROT.

	7	6	5	4	3	2	1	0
R	FPS7	FPS6	FPS5	FPS4	FPS3	FPS2	FPS1	FPDIS
W	((1))	(1)	(1)	(1)	(1)	(1)	(1)	(1)

Reset: This register is loaded from nonvolatile location NVPROT during reset.

NOTES:
1. Background commands can be used to change the contents of these bits in FPROT.

Figure 4-8 FLASH Protection Register (FPROT)

Table 4-10 FPROT Register Field Descriptions

Field	Description
7:1 FPS[7:1]	**FLASH Protect Select Bits** — When FPDIS = 0, this 7-bit field determines the ending address of unprotected FLASH locations at the high address end of the FLASH. Protected FLASH locations cannot be erased or programmed.
0 FPDIS	**FLASH Protection Disable** 0 FLASH block specified by FPS[7:1] is block protected (program and erase not allowed) 1 No FLASH block is protected

4.8.5 FLASH Status Register (FSTAT)

Bits 3, 1, and 0 always read 0 and writes have no meaning or effect. The remaining five bits are status bits that can be read at any time. Writes to these bits have special meanings that are discussed in the bit descriptions.

	7	6	5	4	3	2	1	0
R	FCBEF	FCCF	FPVIOL	FACCERR	0	FBLANK	0	0
W								
Reset:	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-9 FLASH Status Register (FSTAT)

Table 4-11 FSTAT Register Field Descriptions

Field	Description
7 FCBEF	**FLASH Command Buffer Empty Flag** — The FCBEF bit is used to launch commands. It also indicates that the command buffer is empty so that a new command sequence can be executed when performing burst programming. The FCBEF bit is cleared by writing a one to it or when a burst program command is transferred to the array for programming. Only burst program commands can be buffered. 0 Command buffer is full (not ready for additional commands) 1 A new burst program command can be written to the command buffer
6 FCCF	**FLASH Command Complete Flag** — FCCF is set automatically when the command buffer is empty and no command is being processed. FCCF is cleared automatically when a new command is started (by writing 1 to FCBEF to register a command). Writing to FCCF has no meaning or effect. 0 Command in progress 1 All commands complete
5 FPVIOL	**Protection Violation Flag** — FPVIOL is set automatically when FCBEF is cleared to register a command that attempts to erase or program a location in a protected block (the erroneous command is ignored). FPVIOL is cleared by writing a 1 to FPVIOL. 0 No protection violation 1 An attempt was made to erase or program a protected location
4 FACCERR	**Access Error Flag** — FACCERR is set automatically when the proper command sequence is not obeyed exactly (the erroneous command is ignored), if a program or erase operation is attempted before the FCDIV register has been initialized, or if the MCU enters stop while a command was in progress. For a more detailed discussion of the exact actions that are considered access errors, see Section 4.6.5. FACCERR is cleared by writing a 1 to FACCERR. Writing a 0 to FACCERR has no meaning or effect. 0 No access error 1 An access error has occurred
2 FBLANK	**FLASH Verified as All Blank (erased) Flag** — FBLANK is set automatically at the conclusion of a blank check command if the entire FLASH array was verified to be erased. FBLANK is cleared by clearing FCBEF to write a new valid command. Writing to FBLANK has no meaning or effect. 0 After a blank check command is completed and FCCF = 1, FBLANK = 0 indicates the FLASH array is not completely erased 1 After a blank check command is completed and FCCF = 1, FBLANK = 1 indicates the FLASH array is completely erased (all 0xFF)

4.8.6   FLASH Command Register (FCMD)

Only five command codes are recognized in normal user modes as shown in Table 4-12. Refer to Section 4.6.3, for a detailed discussion of FLASH programming and erase operations.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	FCMD7	FCMD6	FCMD5	FCMD4	FCMD3	FCMD2	FCMD1	FCMD0
Reset:	0	0	0	0	0	0	0	0

Figure 4-10 FLASH Command Register (FCMD)

Table 4-12 FLASH Commands

Command	FCMD	Equate File Label
Blank check	0x05	mBlank
Byte program	0x20	mByteProg
Byte program — burst mode	0x25	mBurstProg
Page erase (512 bytes/page)	0x40	mPageErase
Mass erase (all FLASH)	0x41	mMassErase

All other command codes are illegal and generate an access error.

It is not necessary to perform a blank check command after a mass erase operation. Only blank check is required as part of the security unlocking mechanism.

MPXY8300 Series

SECTION 5   RESET, INTERRUPTS AND SYSTEM CONFIGURATION

This section discusses basic reset and interrupt mechanisms and the various sources of reset and interrupts in the MPXY8300 Series. Some interrupt sources from peripheral modules are discussed in greater detail within other sections of this product specification. This section gathers basic information about all reset and interrupt sources in one place for easy reference. A few reset and interrupt sources, including the computer operating properly (COP) watchdog and real-time interrupt (RTI), are not part of on-chip peripheral systems, but are part of the system control logic.

5.1   Features

Reset and interrupt features include:
- Multiple sources of reset for flexible system configuration and reliable operation
- Temperature sensor and temperature restart wake-up
- Reset status register (SRS) to indicate source of most recent reset
- Separate interrupt vectors for each module (reduces polling overhead)

5.2   MCU Reset

Resetting the MCU provides a way to start processing from a known set of initial conditions. During reset, most control and status registers are forced to initial values and the program counter is loaded from the reset vector ($DFFE:$DFFF). On-chip peripheral modules are disabled and any I/O pins are initially configured as general-purpose high-impedance inputs with any pull-up devices disabled. The I bit in the condition code register (CCR) is set to block maskable interrupts so the user program has a chance to initialize the stack pointer (SP) and system control settings. The SP is forced to $00FF at reset.

The device has seven sources for reset:
- Power-on reset (POR)
- Low-voltage detect (LVD)
- Computer operating properly (COP) timer
- Periodic hardware reset (PRST)
- Illegal opcode detect
- Illegal address detect
- Background debug forced reset

Each of these sources has an associated bit in the system reset status register with the exception of the background debug forced reset and the periodic hardware reset, PRST, that is indicated by the PRF bit in the PWUCS1 register.

5.3   Computer Operating Properly (COP) Watchdog

The COP watchdog is intended to force a system reset when the application software fails to execute as expected. To prevent a system reset from the COP watchdog (when it is enabled), application software must reset the COP timer periodically. If the application program gets lost and fails to reset the COP before it times out, a system reset is generated to force the system back to a known starting point. The COP watchdog is enabled by the COPE bit in SOPT1 register. The COP timer is reset by writing any value to the address of SRS. This write does not affect the data in the read-only SRS. Instead, the act of writing to this address is decoded and sends a reset signal to the COP timer.

The time-out period can be selected by the COPCLKS and the COPT0:2 bits as shown in Table 5-1. The COPCLKS bit selects either the LFO or the CPU bus clock as the clocking source and the COPT0:2 bits select the clock count required for a time-out. The tolerances of these time-out periods is dependent on the selected clock source (LFO or HFO).

Table 5-1 COP Watchdog Time-out Period

COPCLKS	COPT 2	COPT 1	COPT 0	Clock Source	COP Overflow Count	COP Overflow Time (msec, nominal)			
0	0	0	0	LFO	2^5	32			
0	0	0	1	LFO	2^6	64			
0	0	1	0	LFO	2^7	128			
0	0	1	1	LFO	2^8	256			
0	1	0	0	LFO	2^9	512			
0	1	0	1	LFO	2^{10}	1024			
0	1	1	0	LFO	2^{11}	2048			
0	1	1	1	LFO	2^{11}	2048			
						BUSCLKS1:0			
						1:1 (0.5 MHz)	**1:0 (1 MHz)**	**0:1 (2 MHz)**	**0:0 (4MHz)**
1	0	0	0	Bus Clock	2^{13}	16.384	8.192	4.096	2.048
1	0	0	1	Bus Clock	2^{14}	32.768	16.384	8.192	4.096
1	0	1	0	Bus Clock	2^{15}	65.536	32.768	16.384	8.192
1	0	1	1	Bus Clock	2^{16}	131.072	65.536	32.768	16.384
1	1	0	0	Bus Clock	2^{17}	262.144	131.072	65.536	32.768
1	1	0	1	Bus Clock	2^{18}	524.288	262.144	131.072	65.536
1	1	1	0	Bus Clock	2^{19}	1048.576	524.288	262.144	131.072
1	1	1	1	Bus Clock	2^{19}	1048.576	524.288	262.144	131.072

After any reset, the COP watchdog is enabled. This provides a reliable way to detect code that is not executing as intended. If the COP watchdog is not used in an application, it can be disabled by clearing the COPE bit in the write-once SOPT1 register. Even if the application will use the reset default settings in COPE, COPCLKS and COPT0:2, the user should still write to write-once SOPT1 during reset initialization to lock in the settings. That way, they cannot be changed accidentally if the application program gets lost.

The write to SRS that services (clears) the COP timer should not be placed in an interrupt service routine (ISR) because the ISR could continue to be executed periodically even if the main application program fails.

When the MCU is in active background debug mode, the COP watchdog is temporarily disabled.

5.4 Interrupts

Interrupts provide a way to save the current CPU status and registers, execute an interrupt service routine (ISR), and then restore the CPU status so processing resumes where it left off before the interrupt. Other than the software interrupt (SWI), which is a program instruction, interrupts are caused by hardware events. The debug module can also generate an SWI under certain circumstances.

If an event occurs in an enabled interrupt source, an associated read-only status flag will become set. The CPU will not respond until and unless the local interrupt enable is a logic 1 to enable the interrupt. The I bit in the CCR must be a logic 0 to allow interrupts. The global interrupt mask (I bit) in the CCR is initially set after reset which masks (prevents) all maskable interrupt sources. The user program initializes the stack pointer and performs other system setup before clearing the I bit to allow the CPU to respond to interrupts. When the CPU receives a qualified interrupt request, it completes the current instruction before responding to the interrupt. The interrupt sequence follows the same cycle-by-cycle sequence as the SWI instruction and consists of:

- Saving the CPU registers on the stack
- Setting the I bit in the CCR to mask further interrupts
- Fetching the interrupt vector for the highest-priority interrupt that is currently pending
- Filling the instruction queue with the first three bytes of program information starting from the address fetched from the interrupt vector locations

While the CPU is responding to the interrupt, the I bit is automatically set to avoid the possibility of another interrupt interrupting the ISR itself (this is called nesting of interrupts). Normally, the I bit is restored to 0 when the CCR is restored from the value stacked on entry to the ISR. In rare cases, the I bit may be cleared inside an ISR (after clearing the status flag that generated the interrupt) so that other interrupts can be serviced without waiting for the first service routine to finish. This practice is not recommended for anyone other than the most experienced programmers because it can lead to subtle program errors that are difficult to debug.

The interrupt service routine ends with a return-from-interrupt (RTI) instruction which restores the CCR, A, X, and PC registers to their pre-interrupt values by reading the previously saved information off the stack.

When two or more interrupts are pending when the I bit is cleared, the highest priority source is serviced first.

NOTE

For compatibility with the M68HC08, the H register is not automatically saved and restored. It is good programming practice to push H onto the stack at the start of the interrupt service routine (ISR) and restore it just before the RTI that is used to return from the ISR.

5.4.1 Interrupt Stack Frame

Figure 5-1 shows the contents and organization of a stack frame. Before the interrupt, the stack pointer (SP) points at the next available byte location on the stack. The current values of CPU registers are stored on the stack starting with the low-order byte of the program counter (PCL) and ending with the CCR. After stacking, the SP points at the next available location on the stack which is the address that is one less than the address where the CCR was saved. The PC value that is stacked is the address of the instruction in the main program that would have executed next if the interrupt had not occurred.

When an RTI instruction is executed, these values are recovered from the stack in reverse order. As part of the RTI sequence, the CPU fills the instruction pipeline by reading three bytes of program information, starting from the PC address just recovered from the stack.

The status flag causing the interrupt must be acknowledged (cleared) before returning from the ISR. Typically, the flag should be cleared at the beginning of the ISR so that if another interrupt is generated by this same source, it will be registered so it can be serviced after completion of the current ISR.
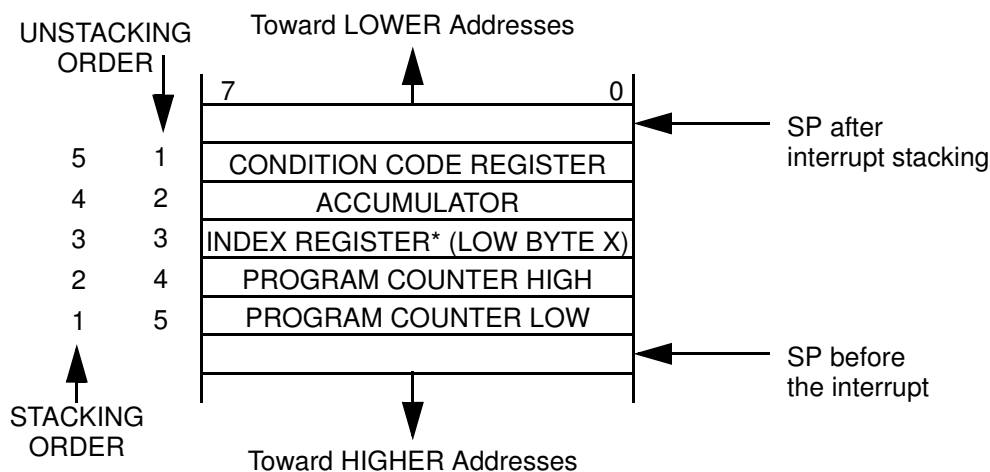
* High byte (H) of index register is not automatically stacked.

Figure 5-1 Interrupt Stack Frame

5.4.2 Interrupt Vectors, Sources and Local Masks

Table 5-2 provides a summary of all interrupt sources. Higher-priority sources are located toward the bottom of the table (at the higher vector addresses). All of these vectors are 2-byte values of the user's destination address. The actual hardware interrupt vectors are located at $FFC0:FFFF. This allows the firmware to intercept all vectors and add additional processing as needed. When an interrupt condition occurs, an associated flag bit becomes set in a globally assigned parameter register at location $005F. The additional process latency and flag bit for each interrupt are described in Codewarrior project files supplied by Freescale.

If the associated local interrupt enable is set, an interrupt request is sent to the CPU. Within the CPU, if the global interrupt mask (I bit in the CCR) is 0, the CPU will finish the current instruction, stack the PCL, PCH, X, A, and CCR CPU registers, set the I bit, and then fetch the interrupt vector for the highest priority pending interrupt. Processing then continues in the interrupt service routine.

The triggering of any of these vector fetches will wake-up the MCU from any of the stop or wait modes.

Table 5-2 Vector Summary

Vector Priority	Vector No.	Vector Addr (High/Low)	Vector Name	Module Source	Flags	Enables	Description
Lower	15	$DFE0:DFE1	Vkbi	KBI	KBF	KBIE	Keyboard interrupt pins PTA2:3, PTA5:6, MISO
	14	$DFE2:DFE3	Vpci	PCI	PCIF	PCIE	Interrupt from the PCI when an automatic pressure reading has been completed.
	13	$DFE4;DFE5	Vacc	ACI	ACIF	ACIE	Interrupt from the ACI when an automatic acceleration reading has been completed
	12	$DFE6:DFE7	Vrti	RTI	RTIF	RTIE	Interrupt from the RTI when the periodic wake-up timer has timed out.
	11	$DFE8:DFE9	Vlfrcvr	LFR	LFCF	LFIE	Interrupt from LFR in carrier detect mode when a carrier present for the required time.
					LFDF		Interrupt from LFR in the manchester decode mode when an 8-bit data byte has been successfully received.
	10	$DFEA:DFEB	Vadc1	ADC10	COCO	AIEN	Interrupt from the ADC10 when a conversion is complete.
	9	$DFEC:DFED	Reserved				
	8	$DFEE:DFEF	Vspi1	SPI	SPRF	SPIE	Interrupt from the SPI when the read data buffer is full.
					MODF		Interrupt from the SPI when there is a mode fault.
					SPTEF	SPTIE	Interrupt from the SPI when the transmit buffer is empty.
	7	$DFF0:DFF1	Vtpm1ovf	TPM1	TOF	TOIE	Interrupt from the TPM1 when the timer overflows.
	6	$DFF2:DFF3	Vtpm1ch1	TPM1	CH1F	CH1IE	Interrupt from the TPM1 when the selected event for channel 1 occurs.
	5	$DFF4:DFF5	Vtpm1ch0	TPM1	CH0F	CH0IE	Interrupt from the TPM1 when the selected event for channel 0 occurs.
	4	$DFF6:DFF7	Vwuktmr	PWU	WUF	PWU*	Interrupt from the PWU when the wake-up time interval has elapsed.
	3	$DFF8:DFF9	Vlvd	LVD	LVDF	LVDIE	Interrupt from the LVD when the supply voltage has dropped below the LVD threshold.
	2	$DFFA:DFFB	Reserved				
	1	$DFFC:DFFD	Vswi	SWI opcode	—	—	Interrupt from the CPU when an SWI instruction has been executed.
Higher	0	$DFFE:DFFF	Vreset	POR	—	—	Reset from power on sequence.
				PRF	PRF	PRST0:5	Reset from PWU when the reset interval elapsed.
				COP	—	COPE	Reset when the COP watchdog times out.
				LVD	—	LVDRE	Reset from the LVD when the supply voltage has dropped below the LVD threshold.
				Temp Restart	—	TRE	Reset when the temperature falls below the temperature restart threshold
				Illegal opcode	—	—	Reset from the CPU when trying to execute an illegal opcode.
				Illegal address	—	—	Reset from the CPU when trying to access an illegal address.

*Please refer to Table 11-3 on page 86.

MPXY8300 Series

5.5 Low-Voltage Detect (LVD) System

The MPXY8300 Series includes a system to detect low voltage conditions in order to protect memory contents and control MCU system states during supply voltage variations. The system is comprised of a power-on reset (POR) circuit and an LVD circuit with a user selectable trip voltage, either high (V_{LVDH}) or low (V_{LVDL}). The LVD circuit is enabled when LVDE in SPMSC1 is high and the trip voltage is selected by LVDV in SPMSC3. The LVD is disabled upon entering any of the stop modes unless the LVDSE bit is set. If LVDSE and LVDE are both set, then the MCU will enter the stop4 mode.

5.5.1 Power-On Reset Operation

When power is initially applied to the device, or when the supply voltage drops below the V_{POR} level, the POR circuit will cause a reset condition. As the supply voltage rises, the LVD circuit will hold the chip in reset until the supply has risen above the V_{LVDL} level. Both the POR bit and the LVD bit in SRS are set following a POR.

5.5.2 LVD Reset Operation

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting LVDRE to 1. After an LVD reset has occurred, the LVD system will hold the device in reset until the supply voltage has risen above the level determined by LVDV. The LVD bit in the SRS register is set following either an LVD reset or POR.

5.5.3 LVD Interrupt Operation

When a low voltage condition is detected and the LVD circuit is configured for interrupt operation (LVDE set, LVDIE set, and LVDRE clear), then LVDF will be set and an LVD interrupt will occur.

5.5.4 Low-Voltage Warning (LVW)

The LVD system has a low voltage warning flag, LVWF, to indicate to the user that the supply voltage is approaching, but is still above, the LVD reset voltage. The LVWF can be reset by writing a logical one to the LVWACK bit. The LVW does not have an interrupt associated with it. There are two user selectable trip voltages for the LVW as selected by LVWV in SPMSC3. The LVWF is set when the supply voltage falls below the selected level and cannot be reset until the supply voltage has risen above the selected level. The threshold for falling and rising differ by a small amount of hysteresis.

5.6 System Clock Control

There are three clock sources within the MPXY8300 series MCU:
- Internal high frequency oscillator (HFO) with nominal frequency of 8 MHz
- Internal medium frequency oscillator (MFO) with nominal frequency of 125 kHz
- Internal low frequency oscillator (LFO) with nominal frequency of 1 kHz

The internal MCU bus clock is derived from the HFO and is 1/2, 1/4, 1/8, or 1/16 of the HFO frequency, as set by the BUSCLKS bits in SOPT2. The MFO feeds the HFO with a reference clock. Both the MFO and HFO are enabled when the MCU is in run or wait mode. Additionally, the MFO can be turned on and off by the LF receiver module (LFR). The LFO is used by the RTI and the PWU and runs continuously in all MCU modes: run, wait, stop4, stop3, stop2, and stop1.

The BUSCLKS control bits select the clock frequency of the HFO as given in Table 5-3. These bits are cleared by any MCU reset.

Table 5-3 HFO Frequency Selections

BUSCLKS1	BUSCLKS0	HFO Frequency (MHz)	CPU Bus Frequency (MHz)
0	0	8	4
0	1	4	2
1	0	2	1
1	1	1	0.5

5.7 Real-Time Interrupt (RTI)

The RTI uses either the internal low frequency oscillator (LFO) or the high frequency oscillator output as its clock source. The RTICLKS bit determines which clock is selected. The RTI can be used as a periodic interrupt in MCU run mode, or can be used as a periodic wake-up from all low power modes. The LFO is always active and cannot be powered off by any software control. The HFO clock cannot be used in any stop mode because it will be disabled.

The SRTISC register includes a read-only status flag, a write-only acknowledge bit, and a 3-bit control value (RTIS2:RTIS1:RTIS0) used to select one of seven wake-up periods between 2 ms and 128 ms. The RTI has a local interrupt enable, RTIE, to allow masking of the real-time interrupt. It can be disabled by writing 0:0:0 to RTIS2:RTIS1:RTIS0 so the timer is disabled and no interrupts will be generated. See Section 5.9.8 for more detail on this register and control bits.

5.8 Temperature Sensor and Restart System

The MPXY8300 Series of devices have two temperature sensing mechanisms. The first is an accurate sensor which is accessible through the ADC10 channel 1 whenever the ADC10 is enabled. The second is a less accurate, very low power sensor which generates a wake-up from either stop1 or stop2 when the temperature drops below its threshold of detection if the TRE bit is set in the SOPT2. This is the temperature restart wake-up which is used as follows:

1. The temperature restart wake-up is enabled by software following detection of an over temperature condition using the temperature sensor connected to the ADC10.
2. The temperature triggered wake-up is enabled by setting the TRE bit in SOPT2.
3. User software instructs the MCU to enter either stop1 or stop2 mode to halt execution during the over temperature condition.
4. When the temperature falls below the temperature restart threshold, T_{RESET}, a wake-up is generated to wake the MCU. Exit from stop1 or stop2 will reset the device.

5.9 Reset, Interrupt, and System Control Registers and Control Bits

This section describes the registers associated with general system control of the MCU.

Refer to the direct-page register summary in Section 4, for the absolute address assignments for all registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

Some control bits in the SOPT1 and SPMSC2 registers are related to modes of operation. Although brief descriptions of these bits are provided here, the related functions are discussed in greater detail in Section 3.

5.9.1 Interrupt Pin Request Status and Control Register (IRQSC)

This direct page register includes two unimplemented bits which always read 0, four read/write bits, one read-only status bit, and one write-only bit. These bits are used to configure the IRQ function, report status, and acknowledge IRQ events.

	7	6	5	4	3	2	1	0
R	0	0	IRQEDG	IRQPE	IRQF	0	IRQIE	IRQMOD
W						IRQACK		
Reset	0	0	0	0	0	0	0	0

☐ = Unimplemented or Reserved

Figure 5-2 Interrupt Request Status and Control Register (IRQSC)

Table 5-4 IRQSC Register Field Descriptions

Field	Description
5 IRQEDG	**Interrupt Request (IRQ) Edge Select** — This read/write control bit is used to select the polarity of edges or levels on the IRQ pin that cause IRQF to be set. The IRQMOD control bit determines whether the IRQ pin is sensitive to both edges and levels or only edges. When the IRQ pin is enabled as the IRQ input and is configured to detect rising edges, the optional pull-up resistor is re-configured as an optional pull-down resistor. 0 IRQ is falling edge or falling edge/low-level sensitive. 1 IRQ is rising edge or rising edge/high-level sensitive.
4 IRQPE	**IRQ Pin Enable** — This read/write control bit enables the IRQ pin function. When this bit is set the IRQ pin can be used as an interrupt request. Also, when this bit is set, either an internal pull-up or an internal pull-down resistor is enabled depending on the state of the IRQMOD bit. 0 IRQ pin function is disabled. 1 IRQ pin function is enabled.
3 IRQF	**IRQ Flag** — This read-only status bit indicates when an interrupt request event has occurred. 0 No IRQ request. 1 IRQ event detected.
2 IRQACK	**IRQ Acknowledge** — This write-only bit is used to acknowledge interrupt request events (write 1 to clear IRQF). Writing 0 has no meaning or effect. Reads always return logic 0. If edge-and-level detection is selected (IRQMOD = 1), IRQF cannot be cleared while the IRQ pin remains at its asserted level.
1 IRQIE	**IRQ Interrupt Enable** — This read/write control bit determines whether IRQ events generate a hardware interrupt request. 0 Hardware interrupt requests from IRQF disabled (use polling). 1 Hardware interrupt requested whenever IRQF = 1.
0 IRQMOD	**IRQ Detection Mode** — This read/write control bit selects either edge-only detection or edge-and-level detection. The IRQEDG control bit determines the polarity of edges and levels that are detected as interrupt request events. 0 IRQ event on falling edges or rising edges only. 1 IRQ event on falling edges and low levels or on rising edges and high levels.

5.9.2   System Reset Status Register (SRS)

This register includes seven read-only status flags to indicate the source of the most recent reset. When a debug host forces reset by writing 1 to BDFR in the SBDFR register, none of the status bits in SRS will be set. Writing any value to this register address clears the COP watchdog timer without affecting the contents of this register. The reset state of these bits depends on what caused the MCU to reset.

	7	6	5	4	3	2	1	0
R	POR	PIN	COP	ILOP	ILAD	PWU	LVD	0
W								
Reset:				Writing any value to SIMRS address clears COP watchdog timer.				
Power-on reset:	1	0	0	0	0	0	1	0
Low-voltage reset:	1	0	0	0	0	0	1	0
Any other reset:	0	(1)	(1)	(1)	(1)	0	0	0

NOTES:
1. Any of these reset sources that are active at the time of reset will cause the corresponding bit(s) to be set; bits corresponding to sources that are not active at the time of reset will be cleared.

Figure 5-3 System Reset Status (SRS)

Table 5-5 SRS Register Field Descriptions

Field	Description
7 POR	**Power-On Reset** — Reset was caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVR) status bit is also set to indicate that the reset occurred while the internal supply was below the LVR threshold. 0 Reset not caused by POR 1 POR caused reset
6 PIN	**External Reset Pin** — Reset was caused by an active-low level on the external reset pin. 0 Reset came from external reset pin 1 Reset not caused by external reset pin
5 COP	**Computer Operating Properly (COP) Watchdog** — Reset was caused by the COP watchdog timer timing out. This reset source may be blocked by COPE = 0. 0 Reset not caused by COP time-out 1 Reset caused by COP time-out
4 ILOP	**Illegal Opcode** — Reset was caused by an attempt to execute an unimplemented or illegal opcode. The STOP instruction is considered illegal if stop is disabled by STOPE = 0 in the SOPT register. The BGND instruction is considered illegal if active background mode is disabled by ENBDM = 0 in the BDCSC register. 0 Reset not caused by an illegal opcode 1 Reset caused by an illegal opcode
3 ILAD	**Illegal Address** — Reset was caused by an attempt to access either data or an instruction at an unimplemented memory address. 0 Reset not caused by an illegal address 1 Reset caused by an illegal address
2 PWU	**Programmable Wakeup** — Reset was caused by a PWU reset in run, wait, stop4, and stop3. After stop2 or stop1 exit, PRF in PWUCSI indicates PWU was the source of a wake-up. 0 Reset not caused by PWU. 1 Reset caused by PWU.
1 LVD	**Low Voltage Detect** — If the LVDRE and LVDSE bits are set and the supply drops below the LVD trip voltage, an LVD reset will occur. This bit is also set by POR. 0 Reset not caused by LVD trip or POR. 1 Reset caused by LVD trip or POR.

5.9.3   System Background Debug Force Reset Register (SBDFR)

This register contains a single write-only control bit. A serial background command such as WRITE_BYTE must be used to write to SBDFR. Attempts to write this register from a user program are ignored. Reads always return 0x00.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR[1]
Reset:	0	0	0	0	0	0	0	0

       = Unimplemented or Reserved

NOTES:
1. BDFR is writable only through serial background debug commands, not from user programs.

Figure 5-4 System Background Debug Force Reset Register (SBDFR)

Table 5-6 SBDFR Register Field Descriptions

Field	Description
0 BDFR	**Background Debug Force Reset** — A serial background command such as WRITE_BYTE may be used to allow an external debug host to force a target system reset. Writing 1 to this bit forces an MCU reset. This bit cannot be written from a user program.

MPXY8300 Series

5.9.4 System Options Register 1 (SOPT1)

SOPT1 is a write-once register so only the first write after reset is honored. Any subsequent attempt to write to SOPT1 (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT1 must be written during the user's reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.
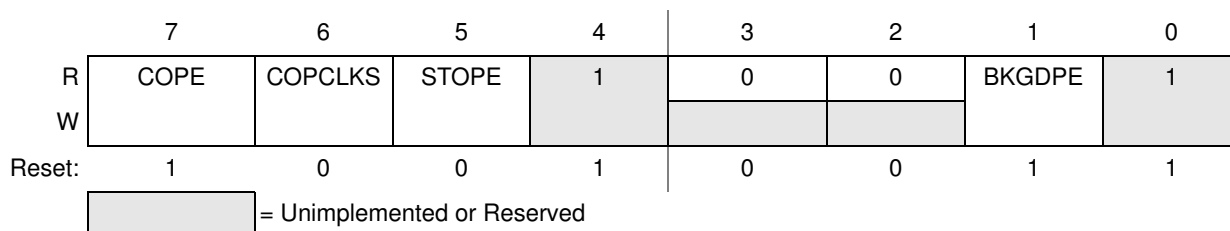
	7	6	5	4	3	2	1	0
R	COPE	COPCLKS	STOPE	1	0	0	BKGDPE	1
W								
Reset:	1	0	0	1	0	0	1	1

☐ = Unimplemented or Reserved

Figure 5-5   System Options Register 1 (SOPT1)

Table 5-7 SOPT1 Register Field Descriptions

Field	Description
7 COPE	**COP Watchdog Enable** — This write-once bit defaults to 1 after reset. 0 COP watchdog timer disabled 1 COP watchdog timer enabled (force reset on time-out)
6 COPCLKS	**COP Clock Selection** — This write-once bit selects the clock source of the COP watchdog. 0 Internal 1-kHz clock is sourced to COP 1 Bus clock is sourced to COP
5 STOPE	**Stop Mode Enable** — This write-once bit defaults to 0 after reset, which disables stop mode. If stop mode is disabled and a user program attempts to execute a STOP instruction, an illegal opcode reset is forced. 0 Stop mode disabled 1 Stop mode enabled
1 BKGDPE	**Background Debug Mode Pin Enable** — The BKGDPE bit enables the PTC0/BKGD/MS pin to function as BKGD/MS. When clear, the pin functions as PTC0, which is an output-only general purpose I/O. This pin defaults to the BKGD/MS function following an MCU reset. 0 PTC0/BKGD/MS pin functions as PTC0 1 PTC0/BKGD/MS pin functions as BKGD/MS

5.9.5 System Options Register 2 (SOPT2)

SOPT2 is a write-once register so only the first write after reset is honored. Any subsequent attempt to write to SOPT2 (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT2 must be written during the user's reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

	7	6	5	4	3	2	1	0
R	TRE		COPT[2:0]		LFOSEL	TCLKDIV	BUSCLKS1	BUSCLKS0
W								
Reset:	0	1	1	1	0	0	0	0

☐ = Unimplemented or Reserved

Figure 5-6   System Options Register 2 (SOPT2)

Table 5-8 SOPT2 Register Field Descriptions

Field	Description
7 TRE	**Temperature Restart Wakeup Enable** — The TRE bit is used to enable the temperature restart wake-up. 0 Temperature restart wake-up disabled 1 Temperature restart wake-up enabled
6:4 COPT[2:0]	**COP Watchdog Time-out —** These write-once bits select the time-out period of the COP. COPT, along with COPCLKS in SOPT1, defines the COP time-out period. See Table 5-1.
3 LFOSEL	**LFO Selected —** LFOSEL determines which signal is connected to the TPM channel 0. 0 TPMEH0 clock input driven by PTA2 1 TPMEH0 clock input driven by LFO
2 TCLKDIV	**Timer Clock Divider** — This write once bit determines the divide ratio of the external clock source, TPMCLK. 0 Divide by 1 1 Divide by 8
1:0 BUSCLKS[1:0]	**Bus Clock Select — HFO divide by:** 00 Divide by 2 01 Divide by 4 10 Divide by 8 11 Divide by 16

5.9.6   System Oscillator Trim Register (SOTRM)

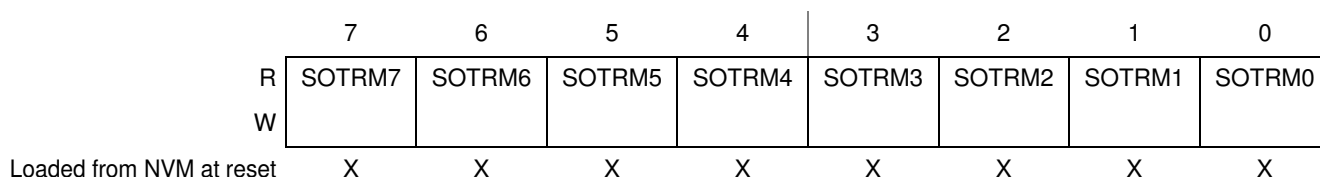This register is used to trim the medium frequency internal oscillator.

	7	6	5	4	3	2	1	0
R	SOTRM7	SOTRM6	SOTRM5	SOTRM4	SOTRM3	SOTRM2	SOTRM1	SOTRM0
W								
Loaded from NVM at reset	X	X	X	X	X	X	X	X

Figure 5-7 System Oscillator Trim Register (SOTRM)

Table 5-9 SOTRM Register Field Descriptions

Field	Description
7:0 SOTRM[7:0]	**MFO Trim Value** — The SOTRM bits are used to trim the frequency of the medium frequency oscillator. Freescale will factory program a FLASH location with this trim value. Upon reset, the FLASH location is read and transferred to this register. Users should not modify this register. Erasure of FLASH will not change the trim value in FLASH.

5.9.7   System Device Identification Register (SDIDH, SDIDL)

This read-only register is included so host development systems can identify the HCS08 derivative and revision number. This allows the development software to recognize where specific memory blocks, registers, and control bits are located in a target MCU.

	7	6	5	4	3	2	1	0
R	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
W								
Reset:	0[1]	0[1]	0[1]	0[1]	0	0	0	0
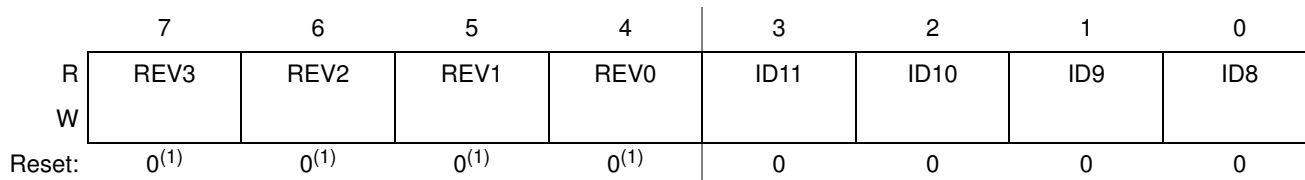
NOTES:
1. The revision number that is hard coded into these bits reflects the current silicon revision level.

Figure 5-8 System Device Identification Register — High (SDIDH)

Table 5-10 SDIDH Register Field Descriptions

Field	Description
7:4 REV[3:0]	**Revision Number** — The high-order four bits of address SDIDH are hard coded to reflect the current mask set revision number (0–F). 0000 0M46E 0001 1M46E 0010 2M46E
3:0 ID[11:8]	**Part Identification Number** — Each derivative in the HCS08 Family has a unique identification number. The MPXY8300 Series is hard coded to the value 0x0010. See Table 5-11 for ID bits in SDIDL.

	7	6	5	4	3	2	1	0
R	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
W								
Reset:	0	0	0	1	0	0	0	0
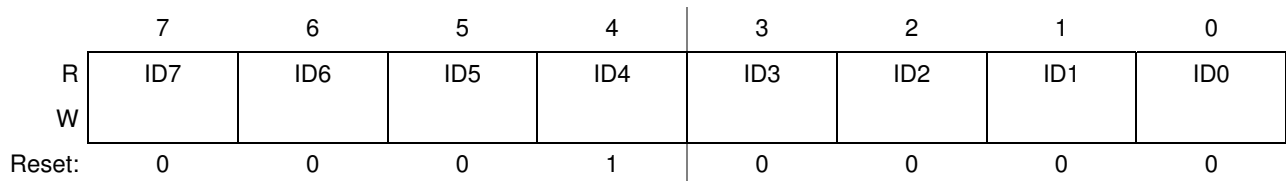
Figure 5-9 System Device Identification Register — Low (SDIDL)

Table 5-11 SDIDL Register Field Descriptions

Field	Description
7:0 ID[7:0]	**Part Identification Number** — Each derivative in the HCS08 Family has a unique identification number. The MPXY8300 Series is hard coded to the value 0x0010. See Table 5-10 for ID bits in SDIDH.

5.9.8   System Real-Time Interrupt Status and Control Register (SRTISC)

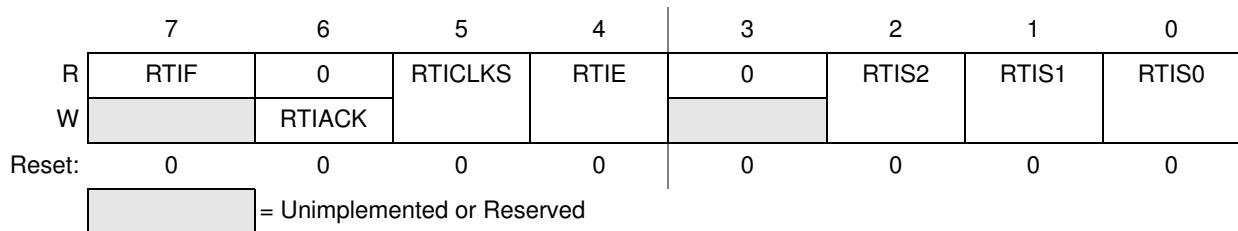This register contains control and status flags related to the real-time interrupt.

	7	6	5	4	3	2	1	0
R	RTIF	0	RTICLKS	RTIE	0	RTIS2	RTIS1	RTIS0
W		RTIACK						
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-10 System RTI Status and Control Register (SRTISC)

Table 5-12 SRTISC Register Field Descriptions

Field	Description
7 RTIF	**Real-Time Interrupt Flag** — This read-only status bit indicates the periodic wake-up timer has timed out. 0 Periodic wake-up timer not timed out 1 Periodic wake-up timer timed out
6 RTIACK	**Real-Time Interrupt Acknowledge** — This write-only bit is used to acknowledge real-time interrupt request 0 Writing 0 has no meaning or effect 1 (write 1 to clear RTIF). Reads always return logic 0
5 RTICLKS	**Real-Time Interrupt Clock Select** — This read-write bit selects the clock source for the real-time interrupt request 0 Real-time interrupt request clock source is internal oscillator. 1 Real-time interrupt request clock source is external clock <table><tr><th>BUSCLKS[1:0]</th><th>HDCLK′ Frequency</th></tr><tr><td>00</td><td>~8 MHz</td></tr><tr><td>01</td><td>~4 MHz</td></tr><tr><td>10</td><td>~2 MHz</td></tr><tr><td>11</td><td>~2 MHz</td></tr></table>
4 RTIE	**Real-Time Interrupt Enable** — This read-write bit enables real-time interrupts. 0 Real-time interrupts disabled 1 Real-time interrupts enabled
2:0 RTIS[2:0]	**Real-Time Interrupt Delay Selects** — These read/write bits select the period for the RTI. See Table 5-13.

Table 5-13 Real-Time Interrupt Period

RTIS2:RTIS1:RTIS0	Using Internal 1-kHz Clock Source[1]	Using HFCLK′ Clock Source
0:0:0	Disable RTI	Disabled
0:0:1	2 ms	$2 * t_{HFCLK}$
0:1:0	4 ms	$4 * t_{HFCLK}$
0:1:1	8 ms	$8 * t_{HFCLK}$
1:0:0	16 ms	$16* t_{HFCLK}$
1:0:1	32 ms	$32 * t_{HFCLK}$
1:1:0	64 ms	$64 * t_{HFCLK}$
1:1:1	128 ms	$128 * t_{HFCLK}$

NOTES:
1. Values are shown in this column based on t_{RTI} = 1 ms. See Section 17 t_{RTI} for the tolerance of this value.

MPXY8300 Series

5.9.9 System Power Management Status and Control 1 Register (SPMSC1)

	7	6	5	4	3	2	1[1]	0
R	LVDF	0	LVDIE	LVDRE[2]	LVDSE	LVDE[2]		BGBE
W		LVDACK						
Reset:	0	0	0	1	1	1	0	0

            = Unimplemented or Reserved

NOTES:
1. Bit 1 is a reserved bit that must always be written to 0.
2. This bit can be written only one time after reset. Additional writes are ignored.

Figure 5-11 System Power Management Status and Control 1 Register (SPMSC1)

Table 5-14 SPMSC1 Register Field Descriptions

Field	Description
7 LVDF	**Low-Voltage Detect Flag** — Provided LVDE = 1, this read-only status bit indicates a low-voltage detect event.
6 LVDACK	**Low-Voltage Detect Acknowledge** — This write-only bit is used to acknowledge low voltage detection errors (write 1 to clear LVDF). Reads always return logic 0.
5 LVDIE	**Low-Voltage Detect Interrupt Enable** — This read/write bit enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1
4 LVDRE	**Low-Voltage Detect Reset Enable** — This read/write bit enables LVDF events to generate a hardware reset (provided LVDE = 1). 0 LVDF does not generate hardware resets 1 Force an MCU reset when LVDF = 1
3 LVDSE	**Low-Voltage Detect Stop Enable** — Provided LVDE = 1, this read/write bit determines whether the low-voltage detect function operates when the MCU is in stop mode. 0 Low-voltage detect disabled during stop mode 1 Low-voltage detect enabled during stop mode
2 LVDE	**Low-Voltage Detect Enable** — This read/write bit enables low-voltage detect logic and qualifies the operation of other bits in this register. 0 LVD logic disabled 1 LVD logic enabled
0 BGBE	**Bandgap Buffer Enable** — The BGBE bit is used to enable an internal buffer for the bandgap voltage reference for use by the ADC module on one of its internal channels. 0 Bandgap buffer disabled 1 Bandgap buffer enabled

5.9.10 System Power Management Status and Control 2 Register (SPMSC2)

This register is used to report the status of the low voltage warning function, and to configure the stop mode behavior of the MCU.

	7	6	5	4	3	2	1	0
R	0	0	0	PDF	PPDF	0	PDC[1]	PPDC[1]
W						PPDACK		
Power-on reset:	0	0	0	0	0	0	0	0
Any other reset:	0	0	U	U	0	0	0	0

☐ = Unimplemented or Reserved   U = Unaffected by reset

NOTES:
1. This bit can be written only one time after reset. Additional writes are ignored.

Figure 5-12 System Power Management Status and Control 2 Register (SPMSC2)

Table 5-15 SPMSC2 Register Field Descriptions

Field	Description
4 PDF	**Power Down Flag** — This read-only status bit indicates the MCU has recovered from stop1 mode. 0 MCU has not recovered from stop1 mode 1 MCU recovered from stop1 mode
3 PPDF	**Partial Power Down Flag** — The PPDF bit indicates that the MCU has exited the stop2 mode. 0 Not stop2 mode recovery 1 Stop2 mode recovery
2 PPDACK	**Partial Power Down Acknowledge** — Writing a logic 1 to PPDACK clears the PPDF and the PDF bits
1 PDC	**Power Down Control** — The PDC bit controls entry into the power down (stop2 and stop1) modes 0 Power down modes are disabled 1 Power down modes are enabled
0 PPDC	**Partial Power Down Control** — The PPDC bit controls which power down mode is selected. 0 Stop1 full power down mode enabled if PDC set 1 Stop2 partial power down mode enabled if PDC set

5.9.11 System Power Management Status and Control 3 Register (SPMSC3)

This register contains one read-only status flag, PDF, which indicates that the MCU has exited the full power down mode (stop1 mode). PDF is unaffected by reset and is cleared by writing a logic 1 to the PPDACK bit in SPMSC2.

	7	6	5	4	3	2	1	0
R	LVWF	0	LVDV	LVWV	0	0	0	0
W		LVWACK						
Power-on reset:	0[(1)]	0	0	0	0	0	0	0
LVD reset:	0[1]	0	U	U	0	0	0	0
Any other reset:	0[1]	0	U	U	0	0	0	0

☐ = Unimplemented or Reserved   U = Unaffected by reset

Figure 5-13 System RTI Status and Control Register (SRTISC)

NOTES:
1. LVWF will be set in the case when V_{Supply} transitions below the trip point or after reset and V_{Supply} is already below V_{LVW}.

Table 5-16 SRTISC Register Field Descriptions

Field	Description
7 LVWF	**Low-Voltage Warning Flag** — The LVWF bit indicates the low voltage warning status. 0 Low voltage warning **not** present 1 Low voltage warning is present or was present
6 LVWACK	**Low-Voltage Warning Acknowledge** — The LVWF bit indicates the low voltage warning status. Writing a logic 1 to LVWACK clears LVWF to a logic 0 if a low voltage warning is not present.
5 LVDV	**Low-Voltage Detect Voltage Select** — The LVDV bit selects the LVD trip point voltage (V_{LVD}). 0 Low trip point selected ($V_{LVD} = V_{LVDL}$) 1 High trip point selected ($V_{LVD} = V_{LVDH}$)
4 LVWV	**Low-Voltage Warning Voltage Select** — The LVWV bit selects the LVW trip point voltage (V_{LVW}). 0 Low trip point selected ($V_{LVW} = V_{LVDL}$) 1 High trip point selected ($V_{LVW} = V_{LVDH}$)

5.9.12 System PCI Trim 1 (SPCITRM1)

	7	6	5	4	3	2	1	0
R W	T7	T6	T5	T4	T3	T2	T1	T0
Loaded from NVM at reset	X	X	X	X	X	X	X	X

Figure 5-14 System PCI Trim 1 Register (SPCITRM1)

Table 5-17 SPCITRM1 Register Field Descriptions

Field	Description
7:0 SPCITRM1	**System PCI Trim 1** — Holds the eight lowest trim bits for the PCI. The register is loaded with FLASH contents after any reset. Software is expected to read this data, do appropriate calculations, and write to PCI trim registers.

5.9.13 System PCI Trim 2 (SPCITRM2)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	T10	T9	T8
W								
Loaded from NVM at reset	0	0	0	0	0	X	X	X

Figure 5-15 System PCI Trim 2 Register (SPCITRM2)

Table 5-18 SPCITRM2 Register Field Descriptions

Field	Description
2:0 SPCITRM2	**System PCI Trim 2** — Holds the other three trim bits for the PCI. The register is loaded with FLASH contents after any reset. Software is expected to read this data, do appropriate calculations, and write to PCI trim registers.

SECTION 6   PARALLEL INPUT/OUTPUT CONTROL

This section explains software controls related to parallel input/output (I/O) and pin control. The MPXY8300 Series has one I/O ports which includes a total of 6 general-purpose I/O pins. See Section 2, for more information about pin assignments and external hardware considerations of these pins.

All of these pins are shared with on-chip peripheral functions as shown in Figure 2-1. The peripheral modules have priority over the parallel I/O so that when a peripheral is enabled, the parallel I/O functions associated with the shared pins are disabled. After reset, the shared peripheral functions are disabled so that the pins are controlled by the parallel I/O and are configured as inputs (PTADDn = 0) with pull-up devices disabled (PTAPEn = 0).

NOTE

To avoid extra current drain from floating input pins, the user's reset initialization routine in the application program must either enable on-chip pull-up devices or change the direction of unconnected pins to outputs so the pins do not float.

Reading and writing of parallel I/O is performed through the port data registers. The direction, either input or output, is controlled through the port data direction registers. The parallel I/O port function for an individual pin is illustrated in the block diagram in Figure 1-1.

The data direction control bit (PTADDn) determines whether the output buffer for the associated pin is enabled, and also controls the source for port data register reads. The input buffer for the associated pin is always enabled unless the pin is enabled as an analog function.

When a shared digital function is enabled for a pin, the output buffer is controlled by the shared function. However, the data direction register bit still controls the source for reads of the port data register.

When a shared analog function is enabled for a pin, both the input and output buffers are disabled. A value of 0 is read for any port data bit where the bit is an input (PTADDn = 0) and the input buffer is disabled. In general, whenever a pin is shared with both an alternate digital function and an analog function, the analog function has priority such that if both the digital and analog functions are enabled, the analog function controls the pin.

It is a good programming practice to write to the port data register before changing the direction of a port pin to become an output. This ensures that the pin will not be driven momentarily with an old data value that happened to be in the port data register.

An internal pull-up device can be enabled for each port pin by setting the corresponding bit in one of the pull-up enable registers (PTAPEn). The pull-up device is disabled if the pin is configured as an output by the parallel I/O control logic or any shared peripheral function regardless of the state of the corresponding pull-up enable register bit. The pull-up device is also disabled if the pin is controlled by an analog function.

6.1   Pin Behavior in Stop Modes

Pin behavior following execution of a STOP instruction depends on the stop mode that is entered. An explanation of pin behavior for the various stop modes follows:

- In stop1 mode, all internal registers including general purpose I/O control and data registers are powered off. Each of the pins assumes its default reset state (output buffer and internal pull-up disabled). Upon exit from stop1, all pins must be reconfigured the same as if the MCU had been reset.

- Stop2 mode is a partial power-down mode, whereby I/O latches are maintained in their state as before the STOP instruction was executed. CPU register status and the state of I/O registers must be saved in RAM before the STOP instruction is executed to place the MCU in stop2 mode. Upon recovery from stop2 mode, before accessing any I/O, the user must examine the state of the PPDF bit in the SPMSC2 register. If the PPDF bit is 0, I/O must be initialized as if a power on reset had occurred. If the PPDF bit is 1, I/O data previously stored in RAM, before the STOP instruction was executed, peripherals may require being initialized and restored to their pre-stop condition. The user must then write a 1 to the PPDACK bit in the SPMSC2 register. Access to I/O is now permitted again in the user's application program.

- In stop3 mode, all pin states are maintained because internal logic stays powered up. Upon recovery, all pin functions are the same as before entering stop3.

6.2   Pull Resistor Interaction with KBI Module

Each port pin has a programmable pull-up and pull-down device. The keyboard interrupt module (KBI) works in conjunction with the parallel I/O control registers. Table 6-1 shows the interaction between KBI and I/O registers to control the pull resistors.

Table 6-1 Truth Table for Pull-up and Pull-down Resistors[(1)]

Line No.	PTAPEm	PTADDm	KBIPEn	KBEDGn	Pull-up	Pull-down
1	0	0	0	x	Disabled	Disabled
2	1	0	0	x	Enabled	Disabled
3	x	1	0	x	Disabled	Disabled
4	0	0	1	x	Disabled	Disabled
5	1	0	1	0	Enabled	Disabled
6	1	0	1	1	Disabled	Enabled
7	x	1	1	x	Disabled	Disabled

NOTES:
  1. x = Don't care

6.3   Parallel I/O Registers

This section provides information about the registers associated with the Port A parallel I/O and pin control functions. These parallel I/O registers are located in page zero of the memory map and the pin control registers are located in the high page register section of memory.

Refer to tables in Section 4, for the absolute address assignments for all parallel I/O and pin control registers. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

Port A parallel I/O function is controlled by the registers described in this section.
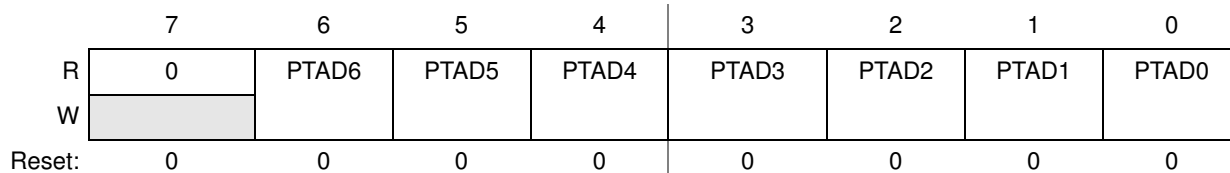
	7	6	5	4	3	2	1	0
R	0	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-1 Port A Data Register (PTAD)

Table 6-2 Port A Data Register Field Descriptions

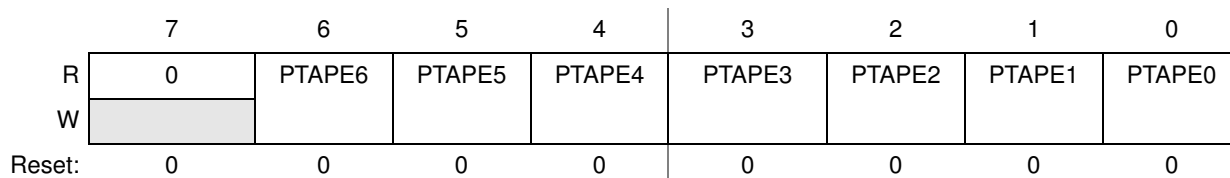| Field	Description
6:0 PTADn	**Port A Data Register Bit n (n = 0–6)** — For port A pins that are inputs, reads return the logic level on the pin. For port A pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port A pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTAD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pull-ups disabled.

	7	6	5	4	3	2	1	0
R	0	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-2 Internal Pull-up Enable for Port A Register (PTAPE)

Table 6-3 Port A Register Field Descriptions

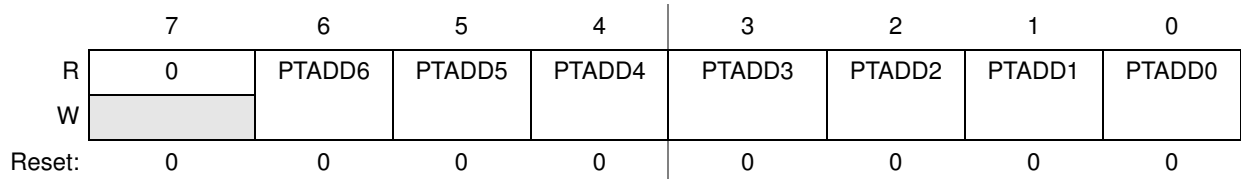| Field	Description
6:0 PTAPEn	**Internal pull-up Enable for Port A Bit n (n = 0–6)** — Each of these control bits determines if the internal pull-up device is enabled for the associated PTA pin. For port A pins that are configured as outputs, these bits have no effect and the internal pull-up devices are disabled. 0 Internal pull-up device disabled for port A bit n 1 Internal pull-up device enabled for port A bit n

	7	6	5	4	3	2	1	0
R	0	PTADD6	PTADD5	PTADD4	PTADD3	PTADD2	PTADD1	PTADD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-3 Data Direction for Port A Register (PTADD)

Table 6-4 Port A Register Field Descriptions

Field	Description
6:0 PTADDn	**Data Direction for Port A Bit n (n = 0–6)** — These read/write bits control the direction of port A pins and what is read for PTAD reads. 0 Input (output driver disabled) and reads return the pin value 1 Output driver enabled for port A bit n and PTAD reads return the contents of PTADn

MPXY8300 Series

SECTION 7   KEYBOARD INTERRUPT

7.1   Introduction

The MPXY8300 Series has a KBI module with two keyboard interrupt inputs that share port A and internal SPI pins.

7.1.1   Features

The KBI features include:
- Up to three keyboard interrupt pins with individual pin enable bits.
- Each keyboard interrupt pin is programmable as falling edge (or rising edge) only, or both falling edge and low level (or both rising edge and high level) interrupt sensitivity.
- One software enabled keyboard interrupt.
- Exit from low-power modes.

7.1.2   Modes of Operation

This section defines the KBI operation in wait, stop, and background debug modes.

7.1.2.1   KBI in Wait Mode

The KBI continues to operate in wait mode if enabled before executing the WAIT instruction. Therefore, an enabled KBI pin (KBPEx = 1) can be used to bring the MCU out of wait mode if the KBI interrupt is enabled (KBIE = 1).

7.1.2.2   KBI in Stop Modes

The KBI operates asynchronously in stop3 mode if enabled before executing the STOP instruction. Therefore, an enabled KBI pin (KBPEx = 1) can be used to bring the MCU out of stop3 mode if the KBI interrupt is enabled (KBIE = 1).

During either stop1 or stop2 mode, the KBI is disabled. In some systems, the pins associated with the KBI may be sources of wake-up from stop1 or stop2, see the stop modes section in the Section 3. Upon wake-up from stop1 or stop2 mode, the KBI module will be in the reset state.

7.1.2.3   KBI in Active Background Mode

When the microcontroller is in active background mode, the KBI will continue to operate normally.

7.1.3   Block Diagram

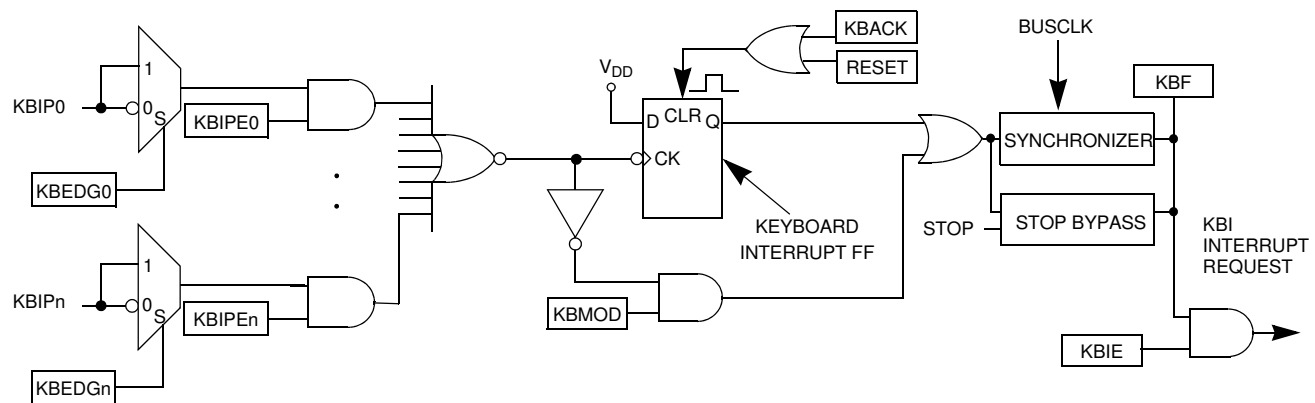The block diagram for the keyboard interrupt module is shown Figure 7-1.

Figure 7-1 KBI Block Diagram

7.2 External Signal Description

The KBI input pins can be used to detect either falling edges, or both falling edge and low level interrupt requests. The KBI input pins can also be used to detect either rising edges, or both rising edge and high level interrupt requests.

The signal properties of KBI are shown in Table 7-1.

Table 7-1 Signal Properties

Signal	Function	I/O
KBIPn	Keyboard interrupt pins	I

7.3 Register Definition

The KBI includes three registers:

- An 8-bit pin status and control register.
- An 8-bit pin enable register.
- An 8-bit edge select register.

Refer to the direct-page register summary in the Section 4 for the absolute address assignments for the KBI registers. This section refers to registers and control bits only by their names and relative address offsets.

7.3.1 KBI Status and Control Register (KBISC)

KBISC contains the status flag and control bits, which are used to configure the KBI.

	7	6	5	4	3	2	1	0
R	0	0	0	0	KBF	0	KBIE	KBMOD
W						KBACK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 7-2 KBI Status and Control Register

Table 7-2 KBISC Register Field Descriptions

Field	Description
7:4	Unused register bits, always read 0.
3 KBF	**Keyboard Interrupt Flag** — KBF indicates when a keyboard interrupt is detected. Writes have no effect on KBF. 0 No keyboard interrupt detected. 1 Keyboard interrupt detected.
2 KBACK	**Keyboard Acknowledge** — Writing a 1 to KBACK is part of the flag clearing mechanism. KBACK always reads as 0.
1 KBIE	**Keyboard Interrupt Enable** — KBIE determines whether a keyboard interrupt is requested. 0 Keyboard interrupt request not enabled. 1 Keyboard interrupt request enabled.
0 KBMOD	**Keyboard Detection Mode** — KBMOD (along with the KBEDG bits) controls the detection mode of the keyboard interrupt pins. 0 Keyboard detects edges only. 1 Keyboard detects both edges and levels.

7.3.2 KBI Pin Enable Register (KBIPE)

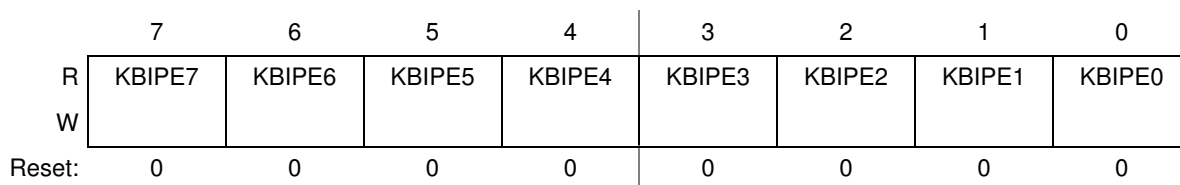KBIPE contains the pin enable control bits.

	7	6	5	4	3	2	1	0
R	KBIPE7	KBIPE6	KBIPE5	KBIPE4	KBIPE3	KBIPE2	KBIPE1	KBIPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 7-3 KBI Pin Enable Register

Table 7-3 KBIPE Register Field Descriptions

Field	Description
7:0 KBIPEn	**Keyboard Pin Enables** — Each of the KBIPEn bits enable the corresponding keyboard interrupt pin. 0 Pin not enabled as keyboard interrupt. 1 Pin enabled as keyboard interrupt.

7.3.3 KBI Edge Select Register (KBIES)

KBIES contains the edge select control bits.

	7	6	5	4	3	2	1	0
R	KBEDG7	KBEDG6	KBEDG5	KBEDG4	KBEDG3	KBEDG2	KBEDG1	KBEDG0
W								
Reset:	0	0	0	0	0	0	0	0
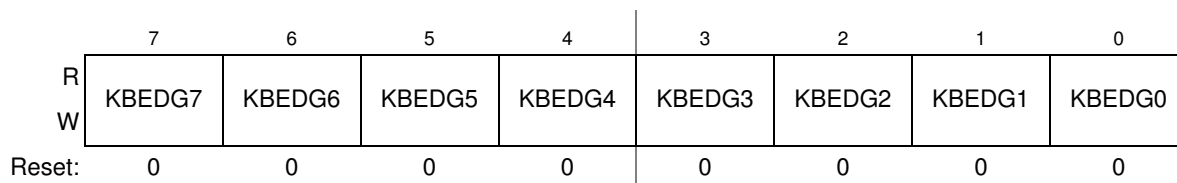
Figure 7-4 KBI Edge Select Register

Table 7-4 KBIES Register Field Descriptions

Field	Description
7:0 KBEDGn	**Keyboard Edge Selects** — Each of the KBEDGn bits selects the falling edge/low level or rising edge/high level function of the corresponding pin). 0 Falling edge/low level. 1 Rising edge/high level.

7.4   Functional Description

This on-chip peripheral module is called a keyboard interrupt (KBI) module because originally it was designed to simplify the connection and use of row-column matrices of keyboard switches. However, these inputs are also useful as extra external interrupt inputs and as an external means of waking the MCU from stop or wait low-power modes.

The KBI module allows up to eight pins to act as additional interrupt sources. Writing to the KBIPEn bits in the keyboard interrupt pin enable register (KBIPE) independently enables or disables each KBI pin. Each KBI pin can be configured as edge sensitive or edge and level sensitive based on the KBMOD bit in the keyboard interrupt status and control register (KBISC). Edge sensitive can be software programmed to be either falling or rising; the level can be either low or high. The polarity of the edge or edge and level sensitivity is selected using the KBEDGn bits in the keyboard interrupt edge select register (KBIES).

Synchronous logic is used to detect edges. Prior to detecting an edge, enabled keyboard inputs must be at the reset logic level. A falling edge is detected when an enabled keyboard input signal is seen as a logic 1 (the reset level) during one bus cycle and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input signal is seen as a logic 0 during one bus cycle and then a logic 1 during the next cycle.

7.4.1 Edge Only Sensitivity

A valid edge on an enabled KBI pin will set KBF in KBISC. If KBIE in KBISC is set, an interrupt request will be presented to the CPU. Clearing of KBF is accomplished by writing a 1 to KBACK in KBISC.

7.4.2 Edge and Level Sensitivity

A valid edge or level on an enabled KBI pin will set KBF in KBISC. If KBIE in KBISC is set, an interrupt request will be presented to the CPU. Clearing of KBF is accomplished by writing a 1 to KBACK in KBISC provided all enabled keyboard inputs are at their reset levels. KBF will remain set if any enabled KBI pin is asserted while attempting to clear by writing a 1 to KBACK.

7.4.3 KBI Pull-up/Pull-down Resistors

The KBI pins can be configured to use an internal pull-up/pull-down resistor using the associated I/O port pull-up enable register. If an internal resistor is enabled, the KBIES register is used to select whether the resistor is a pull-up (KBEDGn = 0) or a pull-down (KBEDGn = 1).

7.4.4 KBI Initialization

When a keyboard interrupt pin is first enabled it is possible to get a false keyboard interrupt flag. To prevent a false interrupt request during keyboard initialization, the user should do the following:

1. Mask keyboard interrupts by clearing KBIE in KBISC.
2. Enable the KBI polarity by setting the appropriate KBEDGn bits in KBIES.
3. If using internal pull-up/pull-down device, configure the associated pull-up enable bits in PTxPE.
4. Enable the KBI pins by setting the appropriate KBIPEn bits in KBIPE.
5. Write to KBACK in KBISC to clear any false interrupts.
6. Set KBIE in KBISC to enable interrupts.

SECTION 8   CENTRAL PROCESSING UNIT

8.1   Introduction

This section provides summary information about the registers, addressing modes, and instruction set of the CPU of the HCS08 Family. For a more detailed discussion, refer to the *HCS08 Family Reference Manual, volume 1,* Freescale Semiconductor document order number HCS08RMV1/D.

The HCS08 CPU is fully source- and object-code-compatible with the M68HC08 CPU. Several instructions and enhanced addressing modes were added to improve C compiler efficiency and to support a new background debug system which replaces the monitor mode of earlier M68HC08 microcontrollers (MCU).

8.1.1   Features

Features of the HCS08 CPU include:
- Object code fully upward-compatible with M68HC05 and M68HC08 Families
- All registers and memory are mapped to a single 64-Kbyte address space
- 16-bit stack pointer (any size stack anywhere in 64-Kbyte address space)
- 16-bit index register (H:X) with powerful indexed addressing modes
- 8-bit accumulator (A)
- Many instructions treat X as a second general-purpose 8-bit register
- Seven addressing modes:
    – Inherent — Operands in internal registers
    – Relative — 8-bit signed offset to branch destination
    – Immediate — Operand in next object code byte(s)
    – Direct — Operand in memory at 0x0000–0x00FF
    – Extended — Operand anywhere in 64-Kbyte address space
    – Indexed relative to H:X — Five submodes including auto increment
    – Indexed relative to SP — Improves C efficiency dramatically
- Memory-to-memory data move instructions with four address mode combinations
- Overflow, half-carry, negative, zero, and carry condition codes support conditional branching on the results of signed, unsigned, and binary-coded decimal (BCD) operations
- Efficient bit manipulation instructions
- Fast 8-bit by 8-bit multiply and 16-bit by 8-bit divide instructions
- STOP and WAIT instructions to invoke low-power operating modes

8.2   Programmer's Model and CPU Registers

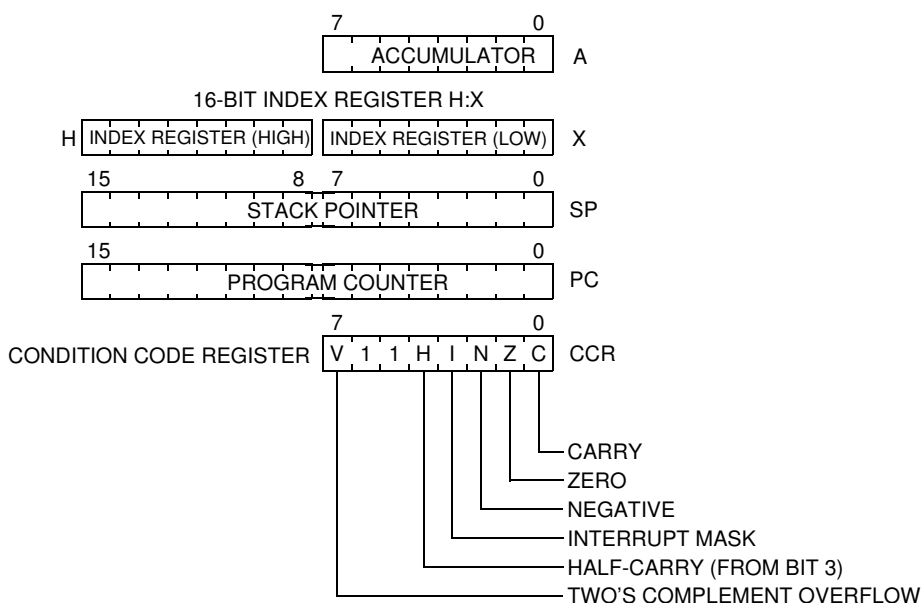Figure 8-1 shows the five CPU registers. CPU registers are not part of the memory map.

Figure 8-1 CPU Registers

8.2.1   Accumulator (A)

The A accumulator is a general-purpose 8-bit register. One operand input to the arithmetic logic unit (ALU) is connected to the accumulator and the ALU results are often stored into the A accumulator after arithmetic and logical operations. The accumulator can be loaded from memory using various addressing modes to specify the address where the loaded data comes from, or the contents of A can be stored to memory using various addressing modes to specify the address where data from A will be stored.

Reset has no effect on the contents of the A accumulator.

8.2.2   Index Register (H:X)

This 16-bit register is actually two separate 8-bit registers (H and X), which often work together as a 16-bit address pointer where H holds the upper byte of an address and X holds the lower byte of the address. All indexed addressing mode instructions use the full 16-bit value in H:X as an index reference pointer; however, for compatibility with the earlier M68HC05 Family, some instructions operate only on the low-order 8-bit half (X).

Many instructions treat X as a second general-purpose 8-bit register that can be used to hold 8-bit data values. X can be cleared, incremented, decremented, complemented, negated, shifted, or rotated. Transfer instructions allow data to be transferred from A or transferred to A where arithmetic and logical operations can then be performed.

For compatibility with the earlier M68HC05 Family, H is forced to 0x00 during reset. Reset has no effect on the contents of X.

8.2.3   Stack Pointer (SP)

This 16-bit address pointer register points at the next available location on the automatic last-in-first-out (LIFO) stack. The stack may be located anywhere in the 64-Kbyte address space that has RAM and can be any size up to the amount of available RAM. The stack is used to automatically save the return address for subroutine calls, the return address and CPU registers during interrupts, and for local variables. The AIS (add immediate to stack pointer) instruction adds an 8-bit signed immediate value to SP. This is most often used to allocate or deallocate space for local variables on the stack.

SP is forced to 0x00FF at reset for compatibility with the earlier M68HC05 Family. HCS08 programs normally change the value in SP to the address of the last location (highest address) in on-chip RAM during reset initialization to free up direct page RAM (from the end of the on-chip registers to 0x00FF).

The RSP (reset stack pointer) instruction was included for compatibility with the M68HC05 Family and is seldom used in new HCS08 programs because it only affects the low-order half of the stack pointer.

8.2.4   Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction or operand to be fetched.

During normal program execution, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, interrupt, and return operations load the program counter with an address other than that of the next sequential location. This is called a change-of-flow.

During reset, the program counter is loaded with the reset vector that is located at 0xFFFE and 0xFFFF. The vector stored there is the address of the first instruction that will be executed after exiting the reset state.

8.2.5   Condition Code Register (CCR)

The 8-bit condition code register contains the interrupt mask (I) and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to 1. The following paragraphs describe the functions of the condition code bits in general terms. For a more detailed explanation of how each instruction sets the CCR bits, refer to the *HCS08 Family Reference Manual, volume 1,* Freescale Semiconductor document order number HCS08RMv1.
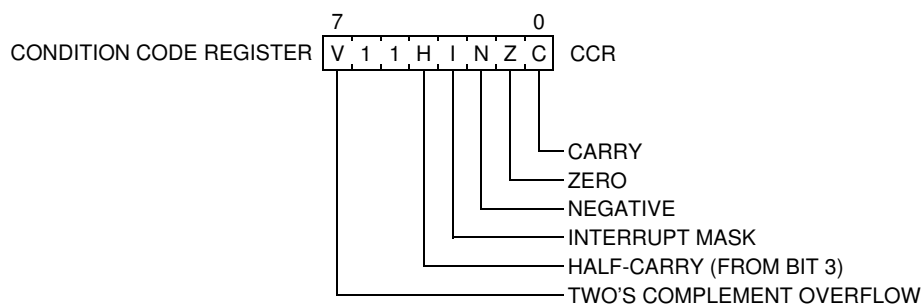

Figure 8-2 Condition Code Register

Table 8-1 CCR Register Field Descriptions

Field	Description
7 V	**Two's Complement Overflow Flag** — The CPU sets the overflow flag when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow flag. 0 No overflow 1 Overflow
4 H	**Half-Carry Flag** — The CPU sets the half-carry flag when a carry occurs between accumulator bits 3 and 4 during an add-without-carry (ADD) or add-with-carry (ADC) operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations. The DAA instruction uses the states of the H and C condition code bits to automatically add a correction value to the result from a previous ADD or ADC on BCD operands to correct the result to a valid BCD value. 0 No carry between bits 3 and 4 1 Carry between bits 3 and 4
3 I	**Interrupt Mask Bit** — When the interrupt mask is set, all maskable CPU interrupts are disabled. CPU interrupts are enabled when the interrupt mask is cleared. When a CPU interrupt occurs, the interrupt mask is set automatically after the CPU registers are saved on the stack, but before the first instruction of the interrupt service routine is executed. Interrupts are not recognized at the instruction boundary after any instruction that clears I (CLI or TAP). This ensures that the next instruction after a CLI or TAP will always be executed without the possibility of an intervening interrupt, provided I was set. 0 Interrupts enabled 1 Interrupts disabled
2 N	**Negative Flag** — The CPU sets the negative flag when an arithmetic operation, logic operation, or data manipulation produces a negative result, setting bit 7 of the result. Simply loading or storing an 8-bit or 16-bit value causes N to be set if the most significant bit of the loaded or stored value was 1. 0 Non-negative result 1 Negative result
1 Z	**Zero Flag** — The CPU sets the zero flag when an arithmetic operation, logic operation, or data manipulation produces a result of 0x00 or 0x0000. Simply loading or storing an 8-bit or 16-bit value causes Z to be set if the loaded or stored value was all 0s. 0 Non-zero result 1 Zero result
0 C	**Carry/Borrow Flag** — The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some instructions — such as bit test and branch, shift, and rotate — also clear or set the carry/borrow flag. 0 No carry out of bit 7 1 Carry out of bit 7

8.3   Addressing Modes

Addressing modes define the way the CPU accesses operands and data. In the HCS08, all memory, status and control registers, and input/output (I/O) ports share a single 64-Kbyte linear address space so a 16-bit binary address can uniquely identify any memory location. This arrangement means that the same instructions that access variables in RAM can also be used to access I/O and control registers or nonvolatile program space.

Some instructions use more than one addressing mode. For instance, move instructions use one addressing mode to specify the source operand and a second addressing mode to specify the destination address. Instructions such as BRCLR, BRSET, CBEQ, and DBNZ use one addressing mode to specify the location of an operand for a test and then use relative addressing mode to specify the branch destination address when the tested condition is true. For BRCLR, BRSET, CBEQ, and DBNZ, the addressing mode listed in the instruction set tables is the addressing mode needed to access the operand to be tested, and relative addressing mode is implied for the branch destination.

8.3.1   Inherent Addressing Mode (INH)

In this addressing mode, operands needed to complete the instruction (if any) are located within CPU registers so the CPU does not need to access memory to get any operands.

8.3.2　Relative Addressing Mode (REL)

Relative addressing mode is used to specify the destination location for branch instructions. A signed 8-bit offset value is located in the memory location immediately following the opcode. During execution, if the branch condition is true, the signed offset is sign-extended to a 16-bit value and is added to the current contents of the program counter, which causes program execution to continue at the branch destination address.

8.3.3　Immediate Addressing Mode (IMM)

In immediate addressing mode, the operand needed to complete the instruction is included in the object code immediately following the instruction opcode in memory. In the case of a 16-bit immediate operand, the high-order byte is located in the next memory location after the opcode, and the low-order byte is located in the next memory location after that.

8.3.4　Direct Addressing Mode (DIR)

In direct addressing mode, the instruction includes the low-order eight bits of an address in the direct page (0x0000–0x00FF). During execution a 16-bit address is formed by concatenating an implied 0x00 for the high-order half of the address and the direct address from the instruction to get the 16-bit address where the desired operand is located. This is faster and more memory efficient than specifying a complete 16-bit address for the operand.

8.3.5　Extended Addressing Mode (EXT)

In extended addressing mode, the full 16-bit address of the operand is located in the next two bytes of program memory after the opcode (high byte first).

8.3.6　Indexed Addressing Mode

Indexed addressing mmode has seven variations including five that use the 16-bit H:X index register pair and two that use the stack pointer as the base reference.

8.3.6.1　Indexed, No Offset (IX)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair as the address of the operand needed to complete the instruction.

8.3.6.2　Indexed, No Offset with Post Increment (IX+)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair as the address of the operand needed to complete the instruction. The index register pair is then incremented (H:X = H:X + 0x0001) after the operand has been fetched. This addressing mode is only used for MOV and CBEQ instructions.

8.3.6.3　Indexed, 8-Bit Offset (IX1)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction.

8.3.6.4　Indexed, 8-Bit Offset with Post Increment (IX1+)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction. The index register pair is then incremented (H:X = H:X + 0x0001) after the operand has been fetched. This addressing mode is used only for the CBEQ instruction.

8.3.6.5　Indexed, 16-Bit Offset (IX2)

This variation of indexed addressing uses the 16-bit value in the H:X index register pair plus a 16-bit offset included in the instruction as the address of the operand needed to complete the instruction.

8.3.6.6　SP-Relative, 8-Bit Offset (SP1)

This variation of indexed addressing uses the 16-bit value in the stack pointer (SP) plus an unsigned 8-bit offset included in the instruction as the address of the operand needed to complete the instruction.

8.3.6.7　SP-Relative, 16-Bit Offset (SP2)

This variation of indexed addressing uses the 16-bit value in the stack pointer (SP) plus a 16-bit offset included in the instruction as the address of the operand needed to complete the instruction.

8.4 Special Operations

The CPU performs a few special operations that are similar to instructions but do not have opcodes like other CPU instructions. In addition, a few instructions such as STOP and WAIT directly affect other MCU circuitry. This section provides additional information about these operations.

8.4.1 Reset Sequence

Reset can be caused by a power-on-reset (POR) event, internal conditions such as the COP (computer operating properly) watchdog, or by assertion of an external active-low reset pin. When a reset event occurs, the CPU immediately stops whatever it is doing (the MCU does not wait for an instruction boundary before responding to a reset event). For a more detailed discussion about how the MCU recognizes resets and determines the source, refer to the Section 5.

The reset event is considered concluded when the sequence to determine whether the reset came from an internal source is done and when the reset pin is no longer asserted. At the conclusion of a reset event, the CPU performs a 6-cycle sequence to fetch the reset vector from 0xFFFE and 0xFFFF and to fill the instruction queue in preparation for execution of the first program instruction.

8.4.2 Interrupt Sequence

When an interrupt is requested, the CPU completes the current instruction before responding to the interrupt. At this point, the program counter is pointing at the start of the next instruction, which is where the CPU should return after servicing the interrupt. The CPU responds to an interrupt by performing the same sequence of operations as for a software interrupt (SWI) instruction, except the address used for the vector fetch is determined by the highest priority interrupt that is pending when the interrupt sequence started.

The CPU sequence for an interrupt is:

1. Store the contents of PCL, PCH, X, A, and CCR on the stack, in that order.
2. Set the I bit in the CCR.
3. Fetch the high-order half of the interrupt vector.
4. Fetch the low-order half of the interrupt vector.
5. Delay for one free bus cycle.
6. Fetch three bytes of program information starting at the address indicated by the interrupt vector to fill the instruction queue in preparation for execution of the first instruction in the interrupt service routine.

After the CCR contents are pushed onto the stack, the I bit in the CCR is set to prevent other interrupts while in the interrupt service routine. Although it is possible to clear the I bit with an instruction in the interrupt service routine, this would allow nesting of interrupts (which is not recommended because it leads to programs that are difficult to debug and maintain).

For compatibility with the earlier M68HC05 MCUs, the high-order half of the H:X index register pair (H) is not saved on the stack as part of the interrupt sequence. The user must use a PSHH instruction at the beginning of the service routine to save H and then use a PULH instruction just before the RTI that ends the interrupt service routine. It is not necessary to save H if you are certain that the interrupt service routine does not use any instructions or auto-increment addressing modes that might change the value of H.

The software interrupt (SWI) instruction is like a hardware interrupt except that it is not masked by the global I bit in the CCR and it is associated with an instruction opcode within the program so it is not asynchronous to program execution.

8.4.3 Wait Mode Operation

The WAIT instruction enables interrupts by clearing the I bit in the CCR. It then halts the clocks to the CPU to reduce overall power consumption while the CPU is waiting for the interrupt or reset event that will wake the CPU from wait mode. When an interrupt or reset event occurs, the CPU clocks will resume and the interrupt or reset event will be processed normally.

If a serial BACKGROUND command is issued to the MCU through the background debug interface while the CPU is in wait mode, CPU clocks will resume and the CPU will enter active background mode where other serial background commands can be processed. This ensures that a host development system can still gain access to a target MCU even if it is in wait mode.

8.4.4 Stop Mode Operation

Usually, all system clocks, including the crystal oscillator (when used), are halted during stop mode to minimize power consumption. In such systems, external circuitry is needed to control the time spent in stop mode and to issue a signal to wake-up the target MCU when it is time to resume processing. Unlike the earlier M68HC05 and M68HC08 MCUs, the HCS08 can be configured to keep a minimum set of clocks running in stop mode. This optionally allows an internal periodic signal to wake the target MCU from stop mode.

When a host debug system is connected to the background debug pin (BKGD) and the ENBDM control bit has been set by a serial command through the background interface (or because the MCU was reset into active background mode), the oscillator is forced to remain active when the MCU enters stop mode. In this case, if a serial BACKGROUND command is issued to the MCU through the background debug interface while the CPU is in stop mode, CPU clocks will resume and the CPU will enter active background mode where other serial background commands can be processed. This ensures that a host development system can still gain access to a target MCU even if it is in stop mode.

Recovery from stop mode depends on the particular HCS08 and whether the oscillator was stopped in stop mode. Refer to the Section 3 for more details.

8.4.5   BGND Instruction

The BGND instruction is new to the HCS08 compared to the M68HC08. BGND would not be used in normal user programs because it forces the CPU to stop processing user instructions and enter the active background mode. The only way to resume execution of the user program is through reset or by a host debug system issuing a GO, TRACE1, or TAGGO serial command through the background debug interface.

Software-based breakpoints can be set by replacing an opcode at the desired breakpoint address with the BGND opcode. When the program reaches this breakpoint address, the CPU is forced to active background mode rather than continuing the user program.

8.5   HCS08 Instruction Set Summary

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in Table 8-2.

Operators

( )	=	Contents of register or memory location shown inside parentheses
←	=	Is loaded with (read: "gets")
&	=	Boolean AND
|	=	Boolean OR
⊕	=	Boolean exclusive-OR
×	=	Multiply
÷	=	Divide
:	=	Concatenate
+	=	Add
−	=	Negate (two's complement)

CPU registers

A	=	Accumulator
CCR	=	Condition code register
H	=	Index register, higher order (most significant) 8 bits
X	=	Index register, lower order (least significant) 8 bits
PC	=	Program counter
PCH	=	Program counter, higher order (most significant) 8 bits
PCL	=	Program counter, lower order (least significant) 8 bits
SP	=	Stack pointer

Memory and addressing

M	=	A memory location or absolute data, depending on addressing mode
M:M + 0x0001	=	A 16-bit value in two consecutive memory locations. The higher-order (most significant) 8 bits are located at the address of M, and the lower-order (least significant) 8 bits are located at the next higher sequential address.

Condition code register (CCR) bits

V	=	Two's complement overflow indicator, bit 7
H	=	Half carry, bit 4
I	=	Interrupt mask, bit 3
N	=	Negative indicator, bit 2
Z	=	Zero indicator, bit 1
C	=	Carry/borrow, bit 0 (carry out of bit 7)

MPXY8300 Series

CCR activity notation

–	=	Bit not affected
0	=	Bit forced to 0
1	=	Bit forced to 1
Þ	=	Bit set or cleared according to results of operation
U	=	Undefined after the operation

Machine coding notation

dd	=	Low-order 8 bits of a direct address 0x0000–0x00FF (high byte assumed to be 0x00)
ee	=	Upper 8 bits of 16-bit offset
ff	=	Lower 8 bits of 16-bit offset or 8-bit offset
ii	=	One byte of immediate data
jj	=	High-order byte of a 16-bit immediate data value
kk	=	Low-order byte of a 16-bit immediate data value
hh	=	High-order byte of 16-bit extended address
ll	=	Low-order byte of 16-bit extended address
rr	=	Relative offset

Source form

Everything in the source forms columns, *except expressions in italic characters,* is literal information that must appear in the assembly source file exactly as shown. The initial 3- to 5-letter mnemonic is always a literal expression. All commas, pound signs (#), parentheses, and plus signs (+) are literal characters.

n	—	Any label or expression that evaluates to a single integer in the range 0–7
opr8i	—	Any label or expression that evaluates to an 8-bit immediate value
opr16i	—	Any label or expression that evaluates to a 16-bit immediate value
opr8a	—	Any label or expression that evaluates to an 8-bit value. The instruction treats this 8-bit value as the low order 8 bits of an address in the direct page of the 64-Kbyte address space (0x00xx).
opr16a	—	Any label or expression that evaluates to a 16-bit value. The instruction treats this value as an address in the 64-Kbyte address space.
oprx8	—	Any label or expression that evaluates to an unsigned 8-bit value, used for indexed addressing
oprx16	—	Any label or expression that evaluates to a 16-bit value. Because the HCS08 has a 16-bit address bus, this can be either a signed or an unsigned value.
rel	—	Any label or expression that refers to an address that is within −128 to +127 locations from the next address after the last byte of object code for the current instruction. The assembler will calculate the 8-bit signed offset and include it in the object code for this instruction.

Address modes

INH	=	Inherent (no operands)
IMM	=	8-bit or 16-bit immediate
DIR	=	8-bit direct
EXT	=	16-bit extended
IX	=	16-bit indexed no offset
IX+	=	16-bit indexed no offset, post increment (CBEQ and MOV only)
IX1	=	16-bit indexed with 8-bit offset from H:X
IX1+	=	16-bit indexed with 8-bit offset, post increment (CBEQ only)
IX2	=	16-bit indexed with 16-bit offset from H:X
REL	=	8-bit relative offset
SP1	=	Stack pointer with 8-bit offset
SP2	=	Stack pointer with 16-bit offset

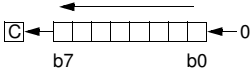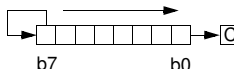**Table 8-2 HCS08 Instruction Set Summary (Sheet 1 of 7)**

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Bus Cycles[1]
ADC #opr8i ADC opr8a ADC opr16a ADC oprx16,X ADC oprx8,X ADC ,X ADC oprx16,SP ADC oprx8,SP	Add with Carry	A ← (A) + (M) + (C)	Þ	Þ	–	Þ	Þ	Þ	IMM DIR EXT IX2 IX1 IX SP2 SP1	A9 B9 C9 D9 E9 F9 9ED9 9EE9	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
ADD #opr8i ADD opr8a ADD opr16a ADD oprx16,X ADD oprx8,X ADD ,X ADD oprx16,SP ADD oprx8,SP	Add without Carry	A ← (A) + (M)	Þ	Þ	–	Þ	Þ	Þ	IMM DIR EXT IX2 IX1 IX SP2 SP1	AB BB CB DB EB FB 9EDB 9EEB	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
AIS #opr8i	Add Immediate Value (Signed) to Stack Pointer	SP ← (SP) + (M) M is sign extended to a 16-bit value	–	–	–	–	–	–	IMM	A7	ii	2
AIX #opr8i	Add Immediate Value (Signed) to Index Register (H:X)	H:X ← (H:X) + (M) M is sign extended to a 16-bit value	–	–	–	–	–	–	IMM	AF	ii	2
AND #opr8i AND opr8a AND opr16a AND oprx16,X AND oprx8,X AND ,X AND oprx16,SP AND oprx8,SP	Logical AND	A ← (A) & (M)	0	–	–	Þ	Þ	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A4 B4 C4 D4 E4 F4 9ED4 9EE4	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
ASL opr8a ASLA ASLX ASL oprx8,X ASL ,X ASL oprx8,SP	Arithmetic Shift Left (Same as LSL)	C ◄─ [◄─[ ][ ][ ][ ][ ][ ][ ][ ]]◄─ 0 b7 b0	Þ	–	–	Þ	Þ	Þ	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	5 1 1 5 4 6
ASR opr8a ASRA ASRX ASR oprx8,X ASR ,X ASR oprx8,SP	Arithmetic Shift Right	[ ][ ][ ][ ][ ][ ][ ][ ] ─► C b7 b0	Þ	–	–	Þ	Þ	Þ	DIR INH INH IX1 IX SP1	37 47 57 67 77 9E67	dd ff ff	5 1 1 5 4 6
BCC rel	Branch if Carry Bit Clear	Branch if (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BCLR n,opr8a	Clear Bit n in Memory	Mn ← 0	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BCS rel	Branch if Carry Bit Set (Same as BLO)	Branch if (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ rel	Branch if Equal	Branch if (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE rel	Branch if Greater Than or Equal To (Signed Operands)	Branch if (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3

MPXY8300 Series

Table 8-2 HCS08 Instruction Set Summary (Sheet 2 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
BGND	Enter Active Background if ENBDM = 1	Waits For and Processes BDM Commands Until GO, TRACE1, or TAGGO	–	–	–	–	–	–	INH	82		5+
BGT rel	Branch if Greater Than (Signed Operands)	Branch if $(Z) \mid (N \oplus V) = 0$	–	–	–	–	–	–	REL	92	rr	3
BHCC rel	Branch if Half Carry Bit Clear	Branch if (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS rel	Branch if Half Carry Bit Set	Branch if (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI rel	Branch if Higher	Branch if $(C) \mid (Z) = 0$	–	–	–	–	–	–	REL	22	rr	3
BHS rel	Branch if Higher or Same (Same as BCC)	Branch if (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH rel	Branch if IRQ Pin High	Branch if IRQ pin = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL rel	Branch if IRQ Pin Low	Branch if IRQ pin = 0	–	–	–	–	–	–	REL	2E	rr	3
BIT #opr8i BIT opr8a BIT opr16a BIT oprx16,X BIT oprx8,X BIT ,X BIT oprx16,SP BIT oprx8,SP	Bit Test	(A) & (M) (CCR Updated but Operands Not Changed)	0	–	–	Þ	Þ	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A5 B5 C5 D5 E5 F5 9ED5 9EE5	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
BLE rel	Branch if Less Than or Equal To (Signed Operands)	Branch if $(Z) \mid (N \oplus V) = 1$	–	–	–	–	–	–	REL	93	rr	3
BLO rel	Branch if Lower (Same as BCS)	Branch if (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS rel	Branch if Lower or Same	Branch if $(C) \mid (Z) = 1$	–	–	–	–	–	–	REL	23	rr	3
BLT rel	Branch if Less Than (Signed Operands)	Branch if $(N \oplus V) = 1$	–	–	–	–	–	–	REL	91	rr	3
BMC rel	Branch if Interrupt Mask Clear	Branch if (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI rel	Branch if Minus	Branch if (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS rel	Branch if Interrupt Mask Set	Branch if (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE rel	Branch if Not Equal	Branch if (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL rel	Branch if Plus	Branch if (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA rel	Branch Always	No Test	–	–	–	–	–	–	REL	20	rr	3
BRCLR n,opr8a,rel	Branch if Bit n in Memory Clear	Branch if (Mn) = 0	–	–	–	–	–	Þ	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	01 03 05 07 09 0B 0D 0F	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5
BRN rel	Branch Never	Uses 3 Bus Cycles	–	–	–	–	–	–	REL	21	rr	3
BRSET n,opr8a,rel	Branch if Bit n in Memory Set	Branch if (Mn) = 1	–	–	–	–	–	Þ	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	00 02 04 06 08 0A 0C 0E	dd rr dd rr dd rr dd rr dd rr dd rr dd rr dd rr	5 5 5 5 5 5 5 5

MPXY8300 Series

Table 8-2 HCS08 Instruction Set Summary (Sheet 3 of 7)

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Bus Cycles[1]
BSET *n*,*opr8a*	Set Bit *n* in Memory	Mn ← 1	–	–	–	–	–	–	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	10 12 14 16 18 1A 1C 1E	dd dd dd dd dd dd dd dd	5 5 5 5 5 5 5 5
BSR *rel*	Branch to Subroutine	PC ← (PC) + 0x0002 push (PCL); SP ← (SP) − 0x0001 push (PCH); SP ← (SP) − 0x0001 PC ← (PC) + *rel*	–	–	–	–	–	–	REL	AD	rr	5
CBEQ *opr8a*,*rel* CBEQA #*opr8i*,*rel* CBEQX #*opr8i*,*rel* CBEQ *oprx8*,X+,*rel* CBEQ ,X+,*rel* CBEQ *oprx8*,SP,*rel*	Compare and Branch if Equal	Branch if (A) = (M) Branch if (A) = (M) Branch if (X) = (M) Branch if (A) = (M) Branch if (A) = (M) Branch if (A) = (M)	–	–	–	–	–	–	DIR IMM IMM IX1+ IX+ SP1	31 41 51 61 71 9E61	dd rr ii rr ii rr ff rr rr ff rr	5 4 4 5 5 6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask Bit	I ← 0	–	–	0	–	–	–	INH	9A		1
CLR *opr8a* CLRA CLRX CLRH CLR *oprx8*,X CLR ,X CLR *oprx8*,SP	Clear	M ← 0x00 A ← 0x00 X ← 0x00 H ← 0x00 M ← 0x00 M ← 0x00 M ← 0x00	0	–	–	0	1	–	DIR INH INH INH IX1 IX SP1	3F 4F 5F 8C 6F 7F 9E6F	dd ff ff	5 1 1 1 5 4 6
CMP #*opr8i* CMP *opr8a* CMP *opr16a* CMP *oprx16*,X CMP *oprx8*,X CMP ,X CMP *oprx16*,SP CMP *oprx8*,SP	Compare Accumulator with Memory	(A) − (M) (CCR Updated But Operands Not Changed)	Þ	–	–	Þ	Þ	Þ	IMM DIR EXT IX2 IX1 IX SP2 SP1	A1 B1 C1 D1 E1 F1 9ED1 9EE1	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
COM *opr8a* COMA COMX COM *oprx8*,X COM ,X COM *oprx8*,SP	Complement (One's Complement)	M ← $(\overline{M})$= 0xFF − (M) A ← $(\overline{A})$ = 0xFF − (A) X ← $(\overline{X})$ = 0xFF − (X) M ← $(\overline{M})$ = 0xFF − (M) M ← $(\overline{M})$ = 0xFF − (M) M ← $(\overline{M})$ = 0xFF − (M)	0	–	–	Þ	Þ	1	DIR INH INH IX1 IX SP1	33 43 53 63 73 9E63	dd ff ff	5 1 1 5 4 6
CPHX *opr16a* CPHX #*opr16i* CPHX *opr8a* CPHX *oprx8*,SP	Compare Index Register (H:X) with Memory	(H:X) − (M:M + 0x0001) (CCR Updated But Operands Not Changed)	Þ	–	–	Þ	Þ	Þ	EXT IMM DIR SP1	3E 65 75 9EF3	hh ll jj kk dd ff	6 3 5 6
CPX #*opr8i* CPX *opr8a* CPX *opr16a* CPX *oprx16*,X CPX *oprx8*,X CPX ,X CPX *oprx16*,SP CPX *oprx8*,SP	Compare X (Index Register Low) with Memory	(X) − (M) (CCR Updated But Operands Not Changed)	Þ	–	–	Þ	Þ	Þ	IMM DIR EXT IX2 IX1 IX SP2 SP1	A3 B3 C3 D3 E3 F3 9ED3 9EE3	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
DAA	Decimal Adjust Accumulator After ADD or ADC of BCD Values	(A)$_{10}$	U	–	–	Þ	Þ	Þ	INH	72		1

MPXY8300 Series

Table 8-2 HCS08 Instruction Set Summary (Sheet 4 of 7)

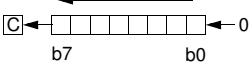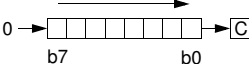| Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Bus Cycles[1]
DBNZ *opr8a,rel* DBNZA *rel* DBNZX *rel* DBNZ *oprx8,X,rel* DBNZ *,X,rel* DBNZ *oprx8,SP,rel*	Decrement and Branch if Not Zero	Decrement A, X, or M Branch if (result) ≠ 0 DBNZX Affects X Not H	–	–	–	–	–	–	DIR INH INH IX1 IX SP1	3B 4B 5B 6B 7B 9E6B	dd rr rr rr ff rr rr ff rr	7 4 4 7 6 8
DEC *opr8a* DECA DECX DEC *oprx8,X* DEC *,X* DEC *oprx8,SP*	Decrement	M ← (M) – 0x01 A ← (A) – 0x01 X ← (X) – 0x01 M ← (M) – 0x01 M ← (M) – 0x01 M ← (M) – 0x01	Þ	–	–	Þ	Þ	–	DIR INH INH IX1 IX SP1	3A 4A 5A 6A 7A 9E6A	dd ff ff	5 1 1 5 4 6
DIV	Divide	A ← (H:A)÷(X) H ← Remainder	–	–	–	–	Þ	Þ	INH	52		6
EOR *#opr8i* EOR *opr8a* EOR *opr16a* EOR *oprx16,X* EOR *oprx8,X* EOR *,X* EOR *oprx16,SP* EOR *oprx8,SP*	Exclusive OR Memory with Accumulator	A ← (A ⊕ M)	0	–	–	Þ	Þ	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A8 B8 C8 D8 E8 F8 9ED8 9EE8	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
INC *opr8a* INCA INCX INC *oprx8,X* INC *,X* INC *oprx8,SP*	Increment	M ← (M) + 0x01 A ← (A) + 0x01 X ← (X) + 0x01 M ← (M) + 0x01 M ← (M) + 0x01 M ← (M) + 0x01	Þ	–	–	Þ	Þ	–	DIR INH INH IX1 IX SP1	3C 4C 5C 6C 7C 9E6C	dd ff ff	5 1 1 5 4 6
JMP *opr8a* JMP *opr16a* JMP *oprx16,X* JMP *oprx8,X* JMP *,X*	Jump	PC ← Jump Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BC CC DC EC FC	dd hh ll ee ff ff	3 4 4 3 3
JSR *opr8a* JSR *opr16a* JSR *oprx16,X* JSR *oprx8,X* JSR *,X*	Jump to Subroutine	PC ← (PC) + *n* (*n* = 1, 2, or 3) Push (PCL); SP ← (SP) – 0x0001 Push (PCH); SP ← (SP) – 0x0001 PC ← Unconditional Address	–	–	–	–	–	–	DIR EXT IX2 IX1 IX	BD CD DD ED FD	dd hh ll ee ff ff	5 6 6 5 5
LDA *#opr8i* LDA *opr8a* LDA *opr16a* LDA *oprx16,X* LDA *oprx8,X* LDA *,X* LDA *oprx16,SP* LDA *oprx8,SP*	Load Accumulator from Memory	A ← (M)	0	–	–	Þ	Þ	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	A6 B6 C6 D6 E6 F6 9ED6 9EE6	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
LDHX *#opr16i* LDHX *opr8a* LDHX *opr16a* LDHX *,X* LDHX *oprx16,X* LDHX *oprx8,X* LDHX *oprx8,SP*	Load Index Register (H:X) from Memory	H:X ← (M:M + 0x0001)	0	–	–	Þ	Þ	–	IMM DIR EXT IX IX2 IX1 SP1	45 55 32 9EAE 9EBE 9ECE 9EFE	jj kk dd hh ll ee ff ff ff	3 4 5 5 6 5 5

Table 8-2 HCS08 Instruction Set Summary (Sheet 5 of 7)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles[1]
			V	H	I	N	Z	C				
LDX #opr8i LDX opr8a LDX opr16a LDX oprx16,X LDX oprx8,X LDX ,X LDX oprx16,SP LDX oprx8,SP	Load X (Index Register Low) from Memory	X ← (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	AE BE CE DE EE FE 9EDE 9EEE	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
LSL opr8a LSLA LSLX LSL oprx8,X LSL ,X LSL oprx8,SP	Logical Shift Left (Same as ASL)		↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	38 48 58 68 78 9E68	dd ff ff	5 1 1 5 4 6
LSR opr8a LSRA LSRX LSR oprx8,X LSR ,X LSR oprx8,SP	Logical Shift Right		↕	–	–	0	↕	↕	DIR INH INH IX1 IX SP1	34 44 54 64 74 9E64	dd ff ff	5 1 1 5 4 6
MOV opr8a,opr8a MOV opr8a,X+ MOV #opr8i,opr8a MOV ,X+,opr8a	Move	(M)$_{destination}$ ← (M)$_{source}$ H:X ← (H:X) + 0x0001 in IX+/DIR and DIR/IX+ Modes	0	–	–	↕	↕	–	DIR/DIR DIR/IX+ IMM/DIR IX+/DIR	4E 5E 6E 7E	dd dd dd ii dd dd	5 5 4 5
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5
NEG opr8a NEGA NEGX NEG oprx8,X NEG ,X NEG oprx8,SP	Negate (Two's Complement)	M ← – (M) = 0x00 – (M) A ← – (A) = 0x00 – (A) X ← – (X) = 0x00 – (X) M ← – (M) = 0x00 – (M) M ← – (M) = 0x00 – (M) M ← – (M) = 0x00 – (M)	↕	–	–	↕	↕	↕	DIR INH INH IX1 IX SP1	30 40 50 60 70 9E60	dd ff ff	5 1 1 5 4 6
NOP	No Operation	Uses 1 Bus Cycle	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap Accumulator	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		1
ORA #opr8i ORA opr8a ORA opr16a ORA oprx16,X ORA oprx8,X ORA ,X ORA oprx16,SP ORA oprx8,SP	Inclusive OR Accumulator and Memory	A ← (A) | (M)	0	–	–	↕	↕	–	IMM DIR EXT IX2 IX1 IX SP2 SP1	AA BA CA DA EA FA 9EDA 9EEA	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
PSHA	Push Accumulator onto Stack	Push (A); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	87		2
PSHH	Push H (Index Register High) onto Stack	Push (H); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	8B		2
PSHX	Push X (Index Register Low) onto Stack	Push (X); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	89		2
PULA	Pull Accumulator from Stack	SP ← (SP + 0x0001); Pull (A)	–	–	–	–	–	–	INH	86		3
PULH	Pull H (Index Register High) from Stack	SP ← (SP + 0x0001); Pull (H)	–	–	–	–	–	–	INH	8A		3
PULX	Pull X (Index Register Low) from Stack	SP ← (SP + 0x0001); Pull (X)	–	–	–	–	–	–	INH	88		3

MPXY8300 Series

Table 8-2 HCS08 Instruction Set Summary (Sheet 6 of 7)
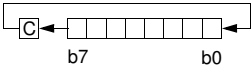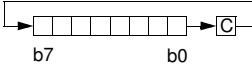
Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Bus Cycles[1]
ROL *opr8a* ROLA ROLX ROL *oprx8*,X ROL ,X ROL *oprx8*,SP	Rotate Left through Carry	C ← [b7 ... b0]	Þ	–	–	Þ	Þ	Þ	DIR INH INH IX1 IX SP1	39 49 59 69 79 9E69	dd ff ff	5 1 1 5 4 6
ROR *opr8a* RORA RORX ROR *oprx8*,X ROR ,X ROR *oprx8*,SP	Rotate Right through Carry	[b7 ... b0] → C	Þ	–	–	Þ	Þ	Þ	DIR INH INH IX1 IX SP1	36 46 56 66 76 9E66	dd ff ff	5 1 1 5 4 6
RSP	Reset Stack Pointer	SP ← 0xFF (High Byte Not Affected)	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 0x0001; Pull (CCR) SP ← (SP) + 0x0001; Pull (A) SP ← (SP) + 0x0001; Pull (X) SP ← (SP) + 0x0001; Pull (PCH) SP ← (SP) + 0x0001; Pull (PCL)	Þ	Þ	Þ	Þ	Þ	Þ	INH	80		9
RTS	Return from Subroutine	SP ← SP + 0x0001; Pull (PCH) SP ← SP + 0x0001; Pull (PCL)	–	–	–	–	–	–	INH	81		6
SBC #*opr8i* SBC *opr8a* SBC *opr16a* SBC *oprx16*,X SBC *oprx8*,X SBC ,X SBC *oprx16*,SP SBC *oprx8*,SP	Subtract with Carry	A ← (A) – (M) – (C)	Þ	–	–	Þ	Þ	Þ	IMM DIR EXT IX2 IX1 IX SP2 SP1	A2 B2 C2 D2 E2 F2 9ED2 9EE2	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask Bit	I ← 1	–	–	1	–	–	–	INH	9B		1
STA *opr8a* STA *opr16a* STA *oprx16*,X STA *oprx8*,X STA ,X STA *oprx16*,SP STA *oprx8*,SP	Store Accumulator in Memory	M ← (A)	0	–	–	Þ	Þ	–	DIR EXT IX2 IX1 IX SP2 SP1	B7 C7 D7 E7 F7 9ED7 9EE7	dd hh ll ee ff ff ee ff ff	3 4 4 3 2 5 4
STHX *opr8a* STHX *opr16a* STHX *oprx8*,SP	Store H:X (Index Reg.)	(M:M + 0x0001) ← (H:X)	0	–	–	Þ	Þ	–	DIR EXT SP1	35 96 9EFF	dd hh ll ff	4 5 5
STOP	Enable Interrupts: Stop Processing Refer to MCU Documentation	I bit ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		2+
STX *opr8a* STX *opr16a* STX *oprx16*,X STX *oprx8*,X STX ,X STX *oprx16*,SP STX *oprx8*,SP	Store X (Low 8 Bits of Index Register) in Memory	M ← (X)	0	–	–	Þ	Þ	–	DIR EXT IX2 IX1 IX SP2 SP1	BF CF DF EF FF 9EDF 9EEF	dd hh ll ee ff ff ee ff ff	3 4 4 3 2 5 4

Table 8-2 HCS08 Instruction Set Summary (Sheet 7 of 7)

Source Form	Operation	Description	V	H	I	N	Z	C	Address Mode	Opcode	Operand	Bus Cycles[1]
SUB #opr8i SUB opr8a SUB opr16a SUB oprx16,X SUB oprx8,X SUB ,X SUB oprx16,SP SUB oprx8,SP	Subtract	A ← (A) − (M)	Þ	–	–	Þ	Þ	Þ	IMM DIR EXT IX2 IX1 IX SP2 SP1	A0 B0 C0 D0 E0 F0 9ED0 9EE0	ii dd hh ll ee ff ff ee ff ff	2 3 4 4 3 3 5 4
SWI	Software Interrupt	PC ← (PC) + 0x0001 Push (PCL); SP ← (SP) − 0x0001 Push (PCH); SP ← (SP) − 0x0001 Push (X); SP ← (SP) − 0x0001 Push (A); SP ← (SP) − 0x0001 Push (CCR); SP ← (SP) − 0x0001 I ← 1; PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		11
TAP	Transfer Accumulator to CCR	CCR ← (A)	Þ	Þ	Þ	Þ	Þ	Þ	INH	84		1
TAX	Transfer Accumulator to X (Index Register Low)	X ← (A)	–	–	–	–	–	–	INH	97		1
TPA	Transfer CCR to Accumulator	A ← (CCR)	–	–	–	–	–	–	INH	85		1
TST opr8a TSTA TSTX TST oprx8,X TST ,X TST oprx8,SP	Test for Negative or Zero	(M) − 0x00 (A) − 0x00 (X) − 0x00 (M) − 0x00 (M) − 0x00 (M) − 0x00	0	–	–	Þ	Þ	–	DIR INH INH IX1 IX SP1	3D 4D 5D 6D 7D 9E6D	dd ff ff	4 1 1 4 3 5
TSX	Transfer SP to Index Reg.	H:X ← (SP) + 0x0001	–	–	–	–	–	–	INH	95		2
TXA	Transfer X (Index Reg. Low) to Accumulator	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer Index Reg. to SP	SP ← (H:X) − 0x0001	–	–	–	–	–	–	INH	94		2
WAIT	Enable Interrupts; Wait for Interrupt	I bit ← 0; Halt CPU	–	–	0	–	–	–	INH	8F		2+

NOTES:
1. Bus clock frequency is one-half of the CPU clock frequency.

Table 8-3 Opcode Map (Sheet 1 of 2)

Bit-Manipulation		Branch	Read-Modify-Write					Control		Register/Memory					
00 5 BRSET0 3 DIR	10 5 BSET0 2 DIR	20 3 BRA 2 REL	30 5 NEG 2 DIR	40 1 NEGA 1 INH	50 1 NEGX 1 INH	60 5 NEG 2 IX1	70 4 NEG 1 IX	80 9 RTI 1 INH	90 3 BGE 2 REL	A0 2 SUB 2 IMM	B0 3 SUB 2 DIR	C0 4 SUB 3 EXT	D0 4 SUB 3 IX2	E0 3 SUB 2 IX1	F0 3 SUB 1 IX
01 5 BRCLR0 3 DIR	11 5 BCLR0 2 DIR	21 3 BRN 2 REL	31 5 CBEQ 3 DIR	41 4 CBEQA 3 IMM	51 4 CBEQX 3 IMM	61 5 CBEQ 3 IX1+	71 5 CBEQ 2 IX+	81 6 RTS 1 INH	91 3 BLT 2 REL	A1 2 CMP 2 IMM	B1 3 CMP 2 DIR	C1 4 CMP 3 EXT	D1 4 CMP 3 IX2	E1 3 CMP 2 IX1	F1 3 CMP 1 IX
02 5 BRSET1 3 DIR	12 5 BSET1 2 DIR	22 3 BHI 2 REL	32 5+ LDHX 3 EXT	42 5 MUL 1 INH	52 6 DIV 1 INH	62 1 NSA 1 INH	72 1 DAA 1 INH	82 BGND 1 INH	92 3 BGT 2 REL	A2 2 SBC 2 IMM	B2 3 SBC 2 DIR	C2 4 SBC 3 EXT	D2 4 SBC 3 IX2	E2 3 SBC 2 IX1	F2 3 SBC 1 IX
03 5 BRCLR1 3 DIR	13 5 BCLR1 2 DIR	23 3 BLS 2 REL	33 5 COM 2 DIR	43 1 COMA 1 INH	53 1 COMX 1 INH	63 5 COM 2 IX1	73 4 COM 1 IX	83 11 SWI 1 INH	93 3 BLE 2 REL	A3 2 CPX 2 IMM	B3 3 CPX 2 DIR	C3 4 CPX 3 EXT	D3 4 CPX 3 IX2	E3 3 CPX 2 IX1	F3 3 CPX 1 IX
04 5 BRSET2 3 DIR	14 5 BSET2 2 DIR	24 3 BCC 2 REL	34 5 LSR 2 DIR	44 1 LSRA 1 INH	54 1 LSRX 1 INH	64 5 LSR 2 IX1	74 4 LSR 1 IX	84 1 TAP 1 INH	94 2 TXS 1 INH	A4 2 AND 2 IMM	B4 3 AND 2 DIR	C4 4 AND 3 EXT	D4 4 AND 3 IX2	E4 3 AND 2 IX1	F4 3 AND 1 IX
05 5 BRCLR2 3 DIR	15 5 BCLR2 2 DIR	25 3 BCS 2 REL	35 4 STHX 2 DIR	45 3 LDHX 3 IMM	55 4 LDHX 2 DIR	65 3 CPHX 3 IMM	75 5 CPHX 2 DIR	85 1 TPA 1 INH	95 2 TSX 1 INH	A5 2 BIT 2 IMM	B5 3 BIT 2 DIR	C5 4 BIT 3 EXT	D5 4 BIT 3 IX2	E5 3 BIT 2 IX1	F5 3 BIT 1 IX
06 5 BRSET3 3 DIR	16 5 BSET3 2 DIR	26 3 BNE 2 REL	36 5 ROR 2 DIR	46 1 RORA 1 INH	56 1 RORX 1 INH	66 5 ROR 2 IX1	76 4 ROR 1 IX	86 3 PULA 1 INH	96 5 STHX 3 EXT	A6 2 LDA 2 IMM	B6 3 LDA 2 DIR	C6 4 LDA 3 EXT	D6 4 LDA 3 IX2	E6 3 LDA 2 IX1	F6 3 LDA 1 IX
07 5 BRCLR3 3 DIR	17 5 BCLR3 2 DIR	27 3 BEQ 2 REL	37 5 ASR 2 DIR	47 1 ASRA 1 INH	57 1 ASRX 1 INH	67 5 ASR 2 IX1	77 4 ASR 1 IX	87 2 PSHA 1 INH	97 1 TAX 1 INH	A7 2 AIS 2 IMM	B7 3 STA 2 DIR	C7 4 STA 3 EXT	D7 4 STA 3 IX2	E7 3 STA 2 IX1	F7 2 STA 1 IX
08 5 BRSET4 3 DIR	18 5 BSET4 2 DIR	28 3 BHCC 2 REL	38 5 LSL 2 DIR	48 1 LSLA 1 INH	58 1 LSLX 1 INH	68 5 LSL 2 IX1	78 4 LSL 1 IX	88 3 PULX 1 INH	98 1 CLC 1 INH	A8 2 EOR 2 IMM	B8 3 EOR 2 DIR	C8 4 EOR 3 EXT	D8 4 EOR 3 IX2	E8 3 EOR 2 IX1	F8 3 EOR 1 IX
09 5 BRCLR4 3 DIR	19 5 BCLR4 2 DIR	29 3 BHCS 2 REL	39 5 ROL 2 DIR	49 1 ROLA 1 INH	59 1 ROLX 1 INH	69 5 ROL 2 IX1	79 4 ROL 1 IX	89 2 PSHX 1 INH	99 1 SEC 1 INH	A9 2 ADC 2 IMM	B9 3 ADC 2 DIR	C9 4 ADC 3 EXT	D9 4 ADC 3 IX2	E9 3 ADC 2 IX1	F9 3 ADC 1 IX
0A 5 BRSET5 3 DIR	1A 5 BSET5 2 DIR	2A 3 BPL 2 REL	3A 5 DEC 2 DIR	4A 1 DECA 1 INH	5A 1 DECX 1 INH	6A 5 DEC 2 IX1	7A 4 DEC 1 IX	8A 3 PULH 1 INH	9A 1 CLI 1 INH	AA 2 ORA 2 IMM	BA 3 ORA 2 DIR	CA 4 ORA 3 EXT	DA 4 ORA 3 IX2	EA 3 ORA 2 IX1	FA 3 ORA 1 IX
0B 5 BRCLR5 3 DIR	1B 5 BCLR5 2 DIR	2B 3 BMI 2 REL	3B 7 DBNZ 3 DIR	4B 4 DBNZA 2 INH	5B 4 DBNZX 2 INH	6B 7 DBNZ 3 IX1	7B 6 DBNZ 2 IX	8B 2 PSHH 1 INH	9B 1 SEI 1 INH	AB 2 ADD 2 IMM	BB 3 ADD 2 DIR	CB 4 ADD 3 EXT	DB 4 ADD 3 IX2	EB 3 ADD 2 IX1	FB 3 ADD 1 IX
0C 5 BRSET6 3 DIR	1C 5 BSET6 2 DIR	2C 3 BMC 2 REL	3C 5 INC 2 DIR	4C 1 INCA 1 INH	5C 1 INCX 1 INH	6C 5 INC 2 IX1	7C 4 INC 1 IX	8C 1 CLRH 1 INH	9C 1 RSP 1 INH	AC (shaded)	BC 3 JMP 2 DIR	CC 4 JMP 3 EXT	DC 4 JMP 3 IX2	EC 3 JMP 2 IX1	FC 3 JMP 1 IX
0D 5 BRCLR6 3 DIR	1D 5 BCLR6 2 DIR	2D 3 BMS 2 REL	3D 4 TST 2 DIR	4D 1 TSTA 1 INH	5D 1 TSTX 1 INH	6D 4 TST 2 IX1	7D 3 TST 1 IX	8D (shaded)	9D 1 NOP 1 INH	AD 5 BSR 2 REL	BD 5 JSR 2 DIR	CD 6 JSR 3 EXT	DD 6 JSR 3 IX2	ED 5 JSR 2 IX1	FD 5 JSR 1 IX
0E 5 BRSET7 3 DIR	1E 5 BSET7 2 DIR	2E 3 BIL 2 REL	3E 6 CPHX 3 EXT	4E 5 MOV 3 DD	5E 5 MOV 2 DIX+	6E 4 MOV 3 IMD	7E 5 MOV 2 IX+D	8E 2+ STOP 1 INH	9E Page 2	AE 2 LDX 2 IMM	BE 3 LDX 2 DIR	CE 4 LDX 3 EXT	DE 4 LDX 3 IX2	EE 3 LDX 2 IX1	FE 3 LDX 1 IX
0F 5 BRCLR7 3 DIR	1F 5 BCLR7 2 DIR	2F 3 BIH 2 REL	3F 5 CLR 2 DIR	4F 1 CLRA 1 INH	5F 1 CLRX 1 INH	6F 5 CLR 2 IX1	7F 4 CLR 1 IX	8F 2+ WAIT 1 INH	9F 1 TXA 1 INH	AF 2 AIX 2 IMM	BF 3 STX 2 DIR	CF 4 STX 3 EXT	DF 4 STX 3 IX2	EF 3 STX 2 IX1	FF 2 STX 1 IX

INH	Inherent	REL	Relative	SP1	Stack Pointer, 8-Bit Offset
IMM	Immediate	IX	Indexed, No Offset	SP2	Stack Pointer, 16-Bit Offset
DIR	Direct	IX1	Indexed, 8-Bit Offset	IX+	Indexed, No Offset with Post Increment
EXT	Extended	IX2	Indexed, 16-Bit Offset		
DD	DIR to DIR	IMD	IMM to DIR	IX1+	Indexed, 1-Byte Offset with Post Increment
IX+D	IX+ to DIR	DIX+	DIR to IX+		

Opcode in Hexadecimal → F0 3 → HCS08 Cycles
SUB → Instruction Mnemonic
Number of Bytes → 1 IX → Addressing Mode

MPXY8300 Series

Table 8-3 Opcode Map (Sheet 2 of 2)

Bit-Manipulation		Branch		Read-Modify-Write				Control				Register/Memory			
				9E60 6 NEG 3 SP1								9ED0 5 SUB 4 SP2	9EE0 4 SUB 3 SP1		
				9E61 6 CBEQ 4 SP1								9ED1 5 CMP 4 SP2	9EE1 4 CMP 3 SP1		
												9ED2 5 SBC 4 SP2	9EE2 4 SBC 3 SP1		
				9E63 6 COM 3 SP1								9ED3 5 CPX 4 SP2	9EE3 4 CPX 3 SP1	9EF3 6 CPHX 3 SP1	
				9E64 6 LSR 3 SP1								9ED4 5 AND 4 SP2	9EE4 4 AND 3 SP1		
												9ED5 5 BIT 4 SP2	9EE5 4 BIT 3 SP1		
				9E66 6 ROR 3 SP1								9ED6 5 LDA 4 SP2	9EE6 4 LDA 3 SP1		
				9E67 6 ASR 3 SP1								9ED7 5 STA 4 SP2	9EE7 4 STA 3 SP1		
				9E68 6 LSL 3 SP1								9ED8 5 EOR 4 SP2	9EE8 4 EOR 3 SP1		
				9E69 6 ROL 3 SP1								9ED9 5 ADC 4 SP2	9EE9 4 ADC 3 SP1		
				9E6A 6 DEC 3 SP1								9EDA 5 ORA 4 SP2	9EEA 4 ORA 3 SP1		
				9E6B 8 DBNZ 4 SP1								9EDB 5 ADD 4 SP2	9EEB 4 ADD 3 SP1		
				9E6C 6 INC 3 SP1											
				9E6D 5 TST 3 SP1											
								9EAE 5 LDHX 2 IX	9EBE 6 LDHX 4 IX2	9ECE 5 LDHX 3 IX1		9EDE 5 LDX 4 SP2	9EEE 4 LDX 3 SP1	9EFE 5 LDHX 3 SP1	
				9E6F 6 CLR 3 SP1								9EDF 5 STX 4 SP2	9EEF 4 STX 3 SP1	9EFF 5 STHX 3 SP1	

INH	Inherent	REL	Relative	SP1	Stack Pointer, 8-Bit Offset
IMM	Immediate	IX	Indexed, No Offset	SP2	Stack Pointer, 16-Bit Offset
DIR	Direct	IX1	Indexed, 8-Bit Offset	IX+	Indexed, No Offset with
EXT	Extended	IX2	Indexed, 16-Bit Offset		Post Increment
DD	DIR to DIR	IMD	IMM to DIR	IX1+	Indexed, 1-Byte Offset with
IX+D	IX+ to DIR	DIX+	DIR to IX+		Post Increment

Note: All Sheet 2 Opcodes are Preceded by the Page 2 Prebyte (9E)

Prebyte (9E) and Opcode in Hexadecimal — 9E60 6 NEG 3 SP1 — HCS08 Cycles, Instruction Mnemonic, Addressing Mode

Number of Bytes

SECTION 9 TIMER/PULSE-WIDTH MODULATOR

The MPXY8300 Series provides one timer/pulse-width modulator (TPM).

9.1 TPM Configuration Information

The device provides one two-channel timer/pulse-width modulator (TPM).

An easy way to measure the low frequency oscillator (LFO) is to connect the LFO directly to TPM channel 0. The LFOSEL bit in the SOPTZ determines whether TPMCH0 is connected to PTAZ or the LFO.

TPM clock source selection for the TPM is shown in the table below.

Table 9-1 TPM Clock Source Selection

CLKSB	CLKSA	Clock Source
0	0	No source; TPM disabled
0	1	BUSCLK
1	0	BUSCLK
1	1	Internal DX pin

9.1.1 Features

The TPM has the following features:
- May be configured for buffered, center-aligned pulse-width modulation (CPWM) on all channels
- Clock sources independently selectable
- Selectable clock sources (device dependent): bus clock, fixed system clock
- Clock prescaler taps for divide by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit free-running or up/down (CPWM) count operation
- 16-bit modulus register to control counter range
- Timer system enable
- One interrupt per channel plus a terminal count interrupt
- Channel features:
    - Each channel may be input capture, output compare, or buffered edge-aligned PWM
    - Rising-edge, falling-edge, or any-edge input capture trigger
    - Set, clear, or toggle output compare action
    - Selectable polarity on PWM outputs

9.1.2   Block Diagram

Figure 9-1 shows the structure of a TPM.
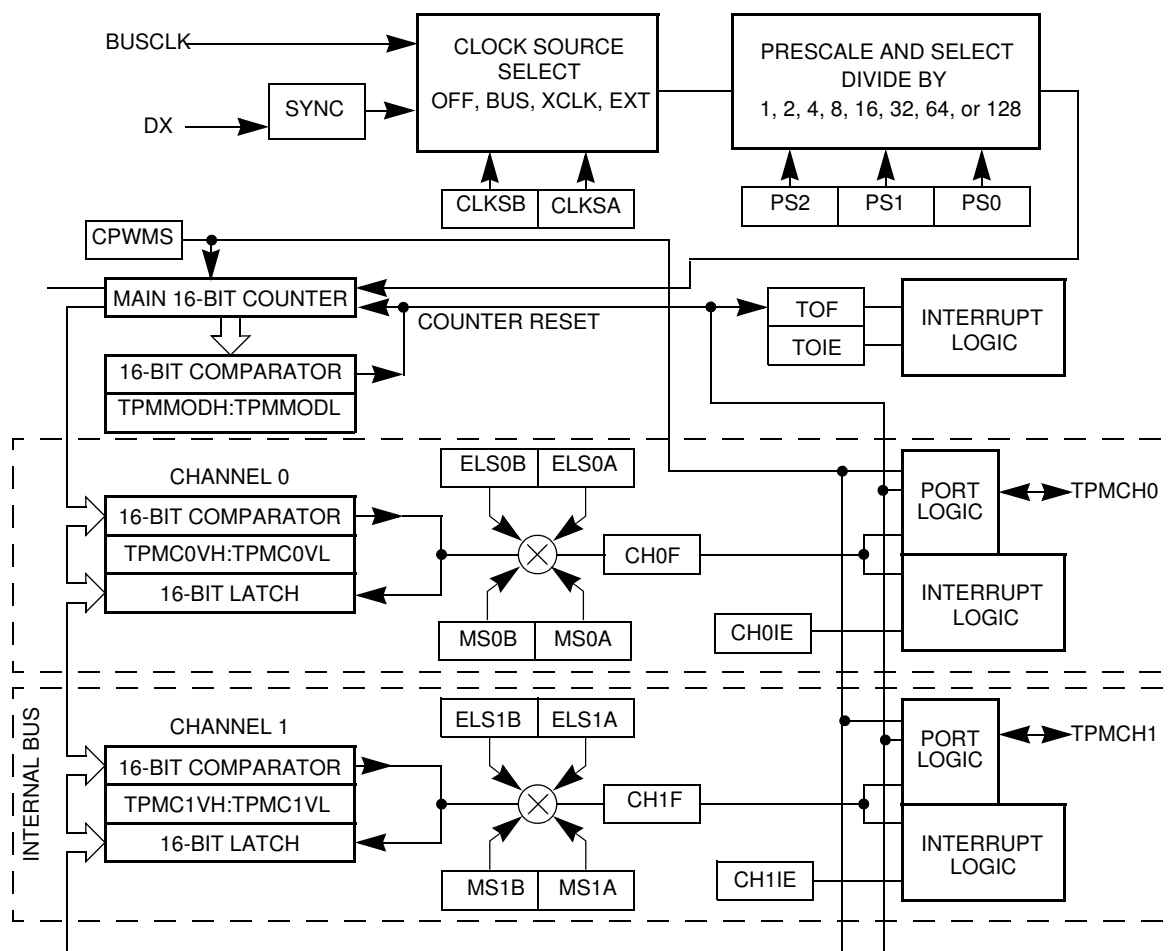
Figure 9-1 TPM Block Diagram

The central component of the TPM is the 16-bit counter that can operate as a free-running counter, a modulo counter, or an up-/down-counter when the TPM is configured for center-aligned PWM. The TPM counter (when operating in normal up-counting mode) provides the timing reference for the input capture, output compare, and edge-aligned PWM functions. The timer counter modulo registers, TPMMODH:TPMMODL, control the modulo value of the counter. (The values 0x0000 or 0xFFFF effectively make the counter free running.) Software can read the counter value at any time without affecting the counting sequence. Any write to either byte of the TPMCNT counter resets the counter regardless of the data value written.

All TPM channels are programmable independently as input capture, output compare, or buffered edge-aligned PWM channels.

9.2   External Signal Description

When any pin associated with the timer is configured as a timer input, a passive pull-up can be enabled. After reset, the TPM modules are disabled and all pins default to general-purpose inputs with the passive pull-ups disabled.

9.2.1   TPMCHn — TPM Channel n I/O Pins

Each TPM channel is associated with an I/O pin on the MCU. The function of this pin depends on the configuration of the channel. In some cases, no pin function is needed so the pin reverts to being controlled by general-purpose I/O controls. When a timer has control of a port pin, the port data and data direction registers do not affect the related pin(s). See the Section 2 for additional information about shared pin functions.

9.3   Register Definition

The TPM includes:

- An 8-bit status and control register (TPMSC)
- A 16-bit counter (TPMCNTH:TPMCNTL)
- A 16-bit modulo register (TPMMODH:TPMMODL)

Each timer channel has:

- An 8-bit status and control register (TPMCnSC)
- A 16-bit channel value register (TPMCnVH:TPMCnVL)

Refer to the direct-page register summary in the Section 4 of this data sheet for the absolute address assignments for all TPM registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

Some MCU systems have more than one TPM, so register names include placeholder characters to identify which TPM and which channel is being referenced. For example, TPMxCnSC refers to timer (TPM) x, channel n and TPM1C2SC is the status and control register for timer 1, channel 2.

9.3.1   Timer x Status and Control Register (TPMSC)

TPMSC contains the overflow status flag and control bits that are used to configure the interrupt enable, TPM configuration, clock source, and prescale divisor. These controls relate to all channels within this timer module.

	7	6	5	4	3	2	1	0
R	TOF	TOIE	CPWMS	CLKSB	CLKSA	PS2	PS1	PS0
W								
Reset	0	0	0	0	0	0	0	0

☐ = Unimplemented or Reserved
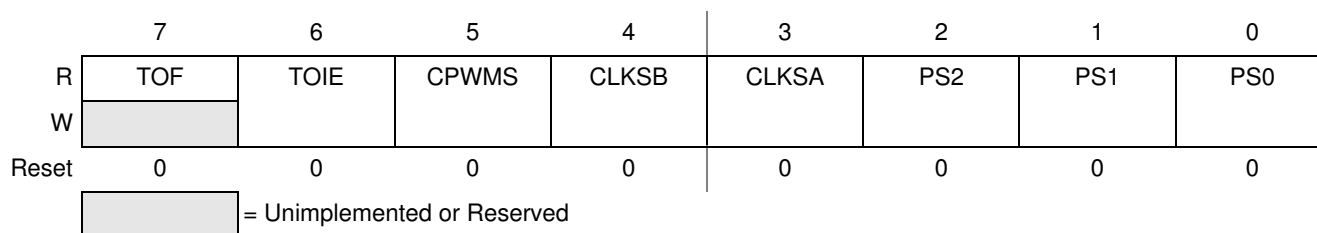
Figure 9-2 Timer x Status and Control Register (TPMSC)

Table 9-2 TPMSC Register Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — This flag is set when the TPM counter changes to 0x0000 after reaching the modulo value programmed in the TPM counter modulo registers. When the TPM is configured for CPWM, TOF is set after the counter has reached the value in the modulo register, at the transition to the next lower count value. Clear TOF by reading the TPM status and control register when TOF is set and then writing a 0 to TOF. If another TPM overflow occurs before the clearing sequence is complete, the sequence is reset so TOF would remain set after the clear sequence was completed for the earlier TOF. Reset clears TOF. Writing a 1 to TOF has no effect. 0 TPM counter has not reached modulo value or overflow 1 TPM counter has overflowed

Table 9-2 TPMSC Register Field Descriptions (Continued)

Field	Description
6 TOIE	**Timer Overflow Interrupt Enable** — This read/write bit enables TPM overflow interrupts. If TOIE is set, an interrupt is generated when TOF equals 1. Reset clears TOIE. 0 TOF interrupts inhibited (use software polling) 1 TOF interrupts enabled
5 CPWMS	**Center-Aligned PWM Select** — This read/write bit selects CPWM operating mode. Reset clears this bit so the TPM operates in up-counting mode for input capture, output compare, and edge-aligned PWM functions. Setting CPWMS reconfigures the TPM to operate in up-/down-counting mode for CPWM functions. Reset clears CPWMS. 0 All TPM channels operate as input capture, output compare, or edge-aligned PWM mode as selected by the MSnB:MSnA control bits in each channel's status and control register 1 All TPM channels operate in center-aligned PWM mode
4:3 CLKS[B:A]	**Clock Source Select** — As shown in Table 9-1, this 2-bit field is used to disable the TPM system or select one of three clock sources to drive the counter prescaler. The internal DX source is synchronized to the bus clock by an on-chip synchronization circuit.
2:0 PS[2:0]	**Prescale Divisor Select** — This 3-bit field selects one of eight divisors for the TPM clock input as shown in Table 9-3. This prescaler is located after any clock source synchronization or clock source selection, so it affects whatever clock source is selected to drive the TPM system.

Table 9-3 Prescale Divisor Selection

PS2:PS1:PS0	TPM Clock Source Divided-By
0:0:0	1
0:0:1	2
0:1:0	4
0:1:1	8
1:0:0	16
1:0:1	32
1:1:0	64
1:1:1	128

9.3.2   Timer x Counter Registers (TPMCNTH:TPMCNTL)

The two read-only TPM counter registers contain the high and low bytes of the value in the TPM counter. Reading either byte (TPMCNTH or TPMCNTL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. This allows coherent 16-bit reads in either order. The coherency mechanism is automatically restarted by an MCU reset, a write of any value to TPMCNTH or TPMCNTL, or any write to the timer status/control register (TPMSC).

Reset clears the TPM counter registers.

	7	6	5	4	3	2	1	0
R	Bit 15	14	13	12	11	10	9	Bit 8
W	Any write to TPMCNTH clears the 16-bit counter.							
Reset	0	0	0	0	0	0	0	0

Figure 9-3 Timer x Counter Register High (TPMCNTH)

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W	Any write to TPMCNTL clears the 16-bit counter.							
Reset	0	0	0	0	0	0	0	0

Figure 9-4 Timer x Counter Register Low (TPMCNTL)

MPXY8300 Series

When background mode is active, the timer counter and the coherency mechanism are frozen such that the buffer latches remain in the state they were in when the background mode became active even if one or both bytes of the counter are read while background mode is active.

9.3.3 Timer x Counter Modulo Registers (TPMMODH:TPMMODL)

The read/write TPM modulo registers contain the modulo value for the TPM counter. After the TPM counter reaches the modulo value, the TPM counter resumes counting from 0x0000 at the next clock (CPWMS = 0) or starts counting down (CPWMS = 1), and the overflow flag (TOF) becomes set. Writing to TPMMODH or TPMMODL inhibits TOF and overflow interrupts until the other byte is written. Reset sets the TPM counter modulo registers to 0x0000, which results in a free-running timer counter (modulo disabled).

	7	6	5	4	3	2	1	0
R	Bit 15	14	13	12	11	10	9	Bit 8
W								
Reset	0	0	0	0	0	0	0	0

Figure 9-5 Timer x Counter Modulo Register High (TPMMODH)

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W								
Reset	0	0	0	0	0	0	0	0

Figure 9-6 Timer x Counter Modulo Register Low (TPMMODL)

It is good practice to wait for an overflow interrupt so both bytes of the modulo register can be written well before a new overflow. An alternative approach is to reset the TPM counter before writing to the TPM modulo registers to avoid confusion about when the first counter overflow will occur.

9.3.4 Timer x Channel n Status and Control Register (TPMCnSC)

TPMCnSC contains the channel interrupt status flag and control bits that are used to configure the interrupt enable, channel configuration, and pin function.

	7	6	5	4	3	2	1	0
R	CHnF	CHnIE	MSnB	MSnA	ELSnB	ELSnA	0	0
W								
Reset	0	0	0	0	0	0	0	0

☐ = Unimplemented or Reserved

Figure 9-7 Timer x Channel n Status and Control Register (TPMCnSC)

Table 9-4 TPMCnSC Register Field Descriptions

Field	Description
7 CHnF	**Channel n Flag** — When channel n is configured for input capture, this flag bit is set when an active edge occurs on the channel n pin. When channel n is an output compare or edge-aligned PWM channel, CHnF is set when the value in the TPM counter registers matches the value in the TPM channel n value registers. This flag is seldom used with center-aligned PWMs because it is set every time the counter matches the channel value register, which correspond to both edges of the active duty cycle period. A corresponding interrupt is requested when CHnF is set and interrupts are enabled (CHnIE = 1). Clear CHnF by reading TPMCnSC while CHnF is set and then writing a 0 to CHnF. If another interrupt request occurs before the clearing sequence is complete, the sequence is reset so CHnF would remain set after the clear sequence was completed for the earlier CHnF. This is done so a CHnF interrupt request cannot be lost by clearing a previous CHnF. Reset clears CHnF. Writing a 1 to CHnF has no effect. 0 No input capture or output compare event occurred on channel n 1 Input capture or output compare event occurred on channel n
6 CHnIE	**Channel n Interrupt Enable** — This read/write bit enables interrupts from channel n. Reset clears CHnIE. 0 Channel n interrupt requests disabled (use software polling) 1 Channel n interrupt requests enabled
5 MSnB	**Mode Select B for TPM Channel n** — When CPWMS = 0, MSnB = 1 configures TPM channel n for edge-aligned PWM mode. For a summary of channel mode and setup controls, refer to Table 9-5.
4 MSnA	**Mode Select A for TPM Channel n** — When CPWMS = 0 and MSnB = 0, MSnA configures TPM channel n for input capture mode or output compare mode. Refer to Table 9-5 for a summary of channel mode and setup controls.
3:2 ELSn[B:A]	**Edge/Level Select Bits** — Depending on the operating mode for the timer channel as set by CPWMS:MSnB:MSnA and shown in Table 9-5, these bits select the polarity of the input edge that triggers an input capture event, select the level that will be driven in response to an output compare match, or select the polarity of the PWM output. Setting ELSnB:ELSnA to 0:0 configures the related timer pin as a general-purpose I/O pin unrelated to any timer channel functions. This function is typically used to temporarily disable an input capture channel or to make the timer pin available as a general-purpose I/O pin when the associated timer channel is set up as a software timer that does not require the use of a pin.

Table 9-5 Mode, Edge, and Level Selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	XX	00	Pin not used for TPM channel; use as an external clock for the TPM or revert to general-purpose I/O	
0	00	01	Input capture	Capture on rising edge only
		10		Capture on falling edge only
		11		Capture on rising or falling edge
	01	00	Output compare	Software compare only
		01		Toggle output on compare
		10		Clear output on compare
		11		Set output on compare
	1X	10	Edge-aligned PWM	High-true pulses (clear output on compare)
		X1		Low-true pulses (set output on compare)
1	XX	10	Center-aligned PWM	High-true pulses (clear output on compare-up)
		X1		Low-true pulses (set output on compare-up)

If the associated port pin is not stable for at least two bus clock cycles before changing to input capture mode, it is possible to get an unexpected indication of an edge trigger. Typically, a program would clear status flags after changing channel configuration bits and before enabling channel interrupts or using the status flags to avoid any unexpected behavior.

9.3.5 Timer x Channel Value Registers (TPMCnVH:TPMCnVL)

These read/write registers contain the captured TPM counter value of the input capture function or the output compare value for the output compare or PWM functions. The channel value registers are cleared by reset.

	7	6	5	4	3	2	1	0
R	Bit 15	14	13	12	11	10	9	Bit 8
W								
Reset	0	0	0	0	0	0	0	0

Figure 9-8 Timer x Channel Value Register High (TPMCnVH)

	7	6	5	4	3	2	1	0
R	Bit 7	6	5	4	3	2	1	Bit 0
W								
Reset	0	0	0	0	0	0	0	0

Figure 9-9 Timer Channel Value Register Low (TPMCnVL)

In input capture mode, reading either byte (TPMCnVH or TPMCnVL) latches the contents of both bytes into a buffer where they remain latched until the other byte is read. This latching mechanism also resets (becomes unlatched) when the TPMCnSC register is written.

In output compare or PWM modes, writing to either byte (TPMCnVH or TPMCnVL) latches the value into a buffer. When both bytes have been written, they are transferred as a coherent 16-bit value into the timer channel value registers. This latching mechanism may be manually reset by writing to the TPMCnSC register.

This latching mechanism allows coherent 16-bit writes in either order, which is friendly to various compiler implementations.

9.4 Functional Description

All TPM functions are associated with a main 16-bit counter that allows flexible selection of the clock source and prescale divisor. A 16-bit modulo register also is associated with the main 16-bit counter in the TPM. Each TPM channel is optionally associated with an MCU pin and a maskable interrupt function.

The TPM has center-aligned PWM capabilities controlled by the CPWMS control bit in TPMSC. When CPWMS is set to 1, timer counter TPMCNT changes to an up-/down-counter and all channels in the associated TPM act as center-aligned PWM channels. When CPWMS = 0, each channel can independently be configured to operate in input capture, output compare, or buffered edge-aligned PWM mode.

The following sections describe the main 16-bit counter and each of the timer operating modes (input capture, output compare, edge-aligned PWM, and center-aligned PWM). Because details of pin operation and interrupt activity depend on the operating mode, these topics are covered in the associated mode sections.

9.4.1 Counter

All timer functions are based on the main 16-bit counter (TPMCNTH:TPMCNTL). This section discusses selection of the clock source, up-counting vs. up-/down-counting, end-of-count overflow, and manual counter reset.

After any MCU reset, CLKSB:CLKSA = 0:0 so no clock source is selected and the TPM is inactive. Normally, CLKSB:CLKSA would be set to 0:1 so the bus clock drives the timer counter. The clock source for each of the TPM can be independently selected to be off, the bus clock (BUSCLK), the fixed system clock (XCLK), or an external input. The maximum frequency allowed for the external clock option is one-fourth the bus rate. Refer to Section 9.3.1 and Table 9-4 for more information about clock source selection.

When the microcontroller is in active background mode, the TPM temporarily suspends all counting until the microcontroller returns to normal user operating mode. During stop mode, all TPM clocks are stopped; therefore, the TPM is effectively disabled until clocks resume. During wait mode, the TPM continues to operate normally.

The main 16-bit counter has two counting modes. When center-aligned PWM is selected (CPWMS = 1), the counter operates in up-/down-counting mode. Otherwise, the counter operates as a simple up-counter. As an up-counter, the main 16-bit counter counts from 0x0000 through its terminal count and then continues with 0x0000. The terminal count is 0xFFFF or a modulus value in TPMMODH:TPMMODL.

When center-aligned PWM operation is specified, the counter counts upward from 0x0000 through its terminal count and then counts downward to 0x0000 where it returns to up-counting. Both 0x0000 and the terminal count value (value in TPMMODH:TPMMODL) are normal length counts (one timer clock period long).

An interrupt flag and enable are associated with the main 16-bit counter. The timer overflow flag (TOF) is a software-accessible indication that the timer counter has overflowed. The enable signal selects between software polling (TOIE = 0) where no hardware interrupt is generated, or interrupt-driven operation (TOIE = 1) where a static hardware interrupt is automatically generated whenever the TOF flag is 1.

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the main 16-bit counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF becomes set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to 0x0000. When the main 16-bit counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and the next lower count value. This corresponds to the end of a PWM period. (The 0x0000 count value corresponds to the center of a period.)

Because the HCS08 MCU is an 8-bit architecture, a coherency mechanism is built into the timer counter for read operations. Whenever either byte of the counter is read (TPMCNTH or TPMCNTL), both bytes are captured into a buffer so when the other byte is read, the value will represent the other byte of the count at the time the first byte was read. The counter continues to count normally, but no new value can be read from either byte until both bytes of the old count have been read.

The main timer counter can be reset manually at any time by writing any value to either byte of the timer count TPMCNTH or TPMCNTL. Resetting the counter in this manner also resets the coherency mechanism in case only one byte of the counter was read before resetting the count.

9.4.2   Channel Mode Selection

Provided CPWMS = 0 (center-aligned PWM operation is not specified), the MSnB and MSnA control bits in the channel n status and control registers determine the basic mode of operation for the corresponding channel. Choices include input capture, output compare, and buffered edge-aligned PWM.

9.4.2.1   Input Capture Mode

With the input capture function, the TPM can capture the time at which an external event occurs. When an active edge occurs on the pin of an input capture channel, the TPM latches the contents of the TPM counter into the channel value registers (TPMCnVH:TPMCnVL). Rising edges, falling edges, or any edge may be chosen as the active edge that triggers an input capture.

When either byte of the 16-bit capture register is read, both bytes are latched into a buffer to support coherent 16-bit accesses regardless of order. The coherency sequence can be manually reset by writing to the channel status/control register (TPMCnSC).

An input capture event sets a flag bit (CHnF) that can optionally generate a CPU interrupt request.

9.4.2.2   Output Compare Mode

With the output compare function, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter reaches the value in the channel value registers of an output compare channel, the TPM can set, clear, or toggle the channel pin.

In output compare mode, values are transferred to the corresponding timer channel value registers only after both 8-bit bytes of a 16-bit register have been written. This coherency sequence can be manually reset by writing to the channel status/control register (TPMCnSC).

An output compare event sets a flag bit (CHnF) that can optionally generate a CPU interrupt request.

9.4.2.3   Edge-Aligned PWM Mode

This type of PWM output uses the normal up-counting mode of the timer counter (CPWMS = 0) and can be used when other channels in the same TPM are configured for input capture or output compare functions. The period of this PWM signal is determined by the setting in the modulus register (TPMMODH:TPMMODL). The duty cycle is determined by the setting in the timer channel value register (TPMCnVH:TPMCnVL). The polarity of this PWM signal is determined by the setting in the ELSnA control bit. Duty cycle cases of 0 percent and 100 percent are possible.

As Figure 9-10 shows, the output compare value in the TPM channel registers determines the pulse width (duty cycle) of the PWM signal. The time between the modulus overflow and the output compare is the pulse width. If ELSnA = 0, the counter overflow forces the PWM signal high and the output compare forces the PWM signal low. If ELSnA = 1, the counter overflow forces the PWM signal low and the output compare forces the PWM signal high.
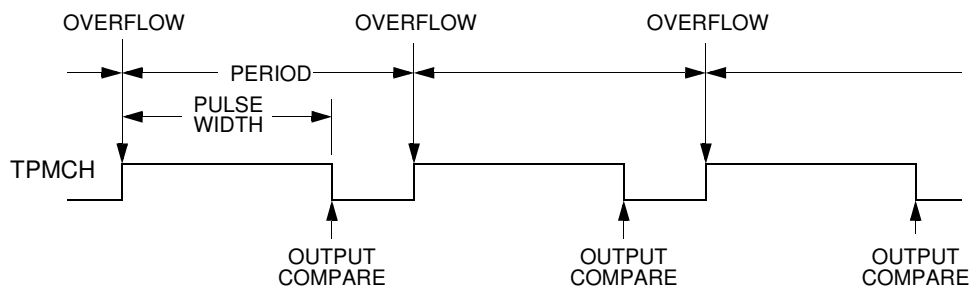

Figure 9-10 PWM Period and Pulse Width (ELSnA = 0)

When the channel value register is set to 0x0000, the duty cycle is 0 percent. By setting the timer channel value register (TPMCnVH:TPMCnVL) to a value greater than the modulus setting, 100% duty cycle can be achieved. This implies that the modulus setting must be less than 0xFFFF to get 100% duty cycle.

Because the HCS08 is a family of 8-bit MCUs, the settings in the timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to either register, TPMCnVH or TPMCnVL, write to buffer registers. In edge-PWM mode, values are transferred to the corresponding timer channel registers only after both 8-bit bytes of a 16-bit register have been written and the value in the TPMCNTH:TPMCNTL counter is 0x0000. (The new duty cycle does not take effect until the next full period.)

9.4.3   Center-Aligned PWM Mode

This type of PWM output uses the up-/down-counting mode of the timer counter (CPWMS = 1). The output compare value in TPMCnVH:TPMCnVL determines the pulse width (duty cycle) of the PWM signal and the period is determined by the value in TPMMODH:TPMMODL. TPMMODH:TPMMODL should be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results. ELSnA will determine the polarity of the CPWM output.

$$\text{pulse width} = 2 \times (\text{TPMCnVH:TPMCnVL})$$

$$\text{period} = 2 \times (\text{TPMMODH:TPMMODL});$$
$$\text{for TPMMODH:TPMMODL} = 0x0001{-}0x7FFF$$

If the channel value register TPMCnVH:TPMCnVL is zero or negative (bit 15 set), the duty cycle will be 0%. If TPMCnVH:TPMCnVL is a positive value (bit 15 clear) and is greater than the (nonzero) modulus setting, the duty cycle will be 100% because the duty cycle compare will never occur. This implies the usable range of periods set by the modulus register is 0x0001 through 0x7FFE (0x7FFF if generation of 100% duty cycle is not necessary). This is not a significant limitation because the resulting period is much longer than required for normal applications.

TPMMODH:TPMMODL = 0x0000 is a special case that should not be used with center-aligned PWM mode. When CPWMS = 0, this case corresponds to the counter running free from 0x0000 through 0xFFFF, but when CPWMS = 1 the counter needs a valid match to the modulus register somewhere other than at 0x0000 in order to change directions from up-counting to down-counting.

Figure 9-11 shows the output compare value in the TPM channel registers (multiplied by 2), which determines the pulse width (duty cycle) of the CPWM signal. If ELSnA = 0, the compare match while counting up forces the CPWM output signal low and a compare match while counting down forces the output high. The counter counts up until it reaches the modulo setting in TPMMODH:TPMMODL, then counts down until it reaches zero. This sets the period equal to two times TPMMODH:TPMMODL.
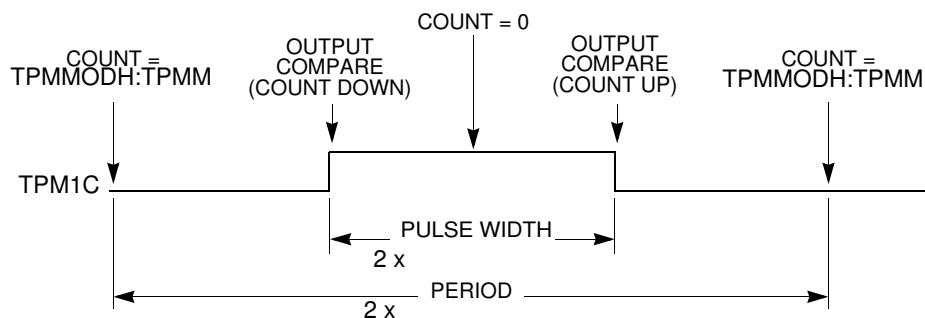
Figure 9-11 CPWM Period and Pulse Width (ELSnA = 0)

Center-aligned PWM outputs typically produce less noise than edge-aligned PWMs because fewer I/O pin transitions are lined up at the same system clock edge. This type of PWM is also required for some types of motor drives.

Because the HCS08 is a family of 8-bit MCUs, the settings in the timer channel registers are buffered to ensure coherent 16-bit updates and to avoid unexpected PWM pulse widths. Writes to any of the registers, TPMMODH, TPMMODL, TPMCnVH, and TPMCnVL, actually write to buffer registers. Values are transferred to the corresponding timer channel registers only after both 8-bit bytes of a 16-bit register have been written and the timer counter overflows (reverses direction from up-counting to down-counting at the end of the terminal count in the modulus register). This TPMCNT overflow requirement only applies to PWM channels, not output compares.

Optionally, when TPMCNTH:TPMCNTL = TPMMODH:TPMMODL, the TPM can generate a TOF interrupt at the end of this count. The user can choose to reload any number of the PWM buffers, and they will all update simultaneously at the start of a new period.

Writing to TPMSC cancels any values written to TPMMODH and/or TPMMODL and resets the coherency mechanism for the modulo registers. Writing to TPMCnSC cancels any values written to the channel value registers and resets the coherency mechanism for TPMCnVH:TPMCnVL.

9.5   TPM Interrupts

The TPM generates an optional interrupt for the main counter overflow and an interrupt for each channel. The meaning of channel interrupts depends on the mode of operation for each channel. If the channel is configured for input capture, the interrupt flag is set each time the selected input capture edge is recognized. If the channel is configured for output compare or PWM modes, the interrupt flag is set each time the main timer counter matches the value in the 16-bit channel value register. See Section 5 for absolute interrupt vector addresses, priority, and local interrupt mask control bits.

For each interrupt source in the TPM, a flag bit is set on recognition of the interrupt condition such as timer overflow, channel input capture, or output compare events. This flag may be read (polled) by software to verify that the action has occurred, or an associated enable bit (TOIE or CHnIE) can be set to enable hardware interrupt generation. While the interrupt enable bit is set, a static interrupt will be generated whenever the associated interrupt flag equals 1. It is the responsibility of user software to perform a sequence of steps to clear the interrupt flag before returning from the interrupt service routine.

9.5.1   Clearing Timer Interrupt Flags

TPM interrupt flags are cleared by a 2-step process that includes a read of the flag bit while it is set (1) followed by a write of 0 to the bit. If a new event is detected between these two steps, the sequence is reset and the interrupt flag remains set after the second step to avoid the possibility of missing the new event.

9.5.2   Timer Overflow Interrupt Description

The conditions that cause TOF to become set depend on the counting mode (up or up/down). In up-counting mode, the 16-bit timer counter counts from 0x0000 through 0xFFFF and overflows to 0x0000 on the next counting clock. TOF becomes set at the transition from 0xFFFF to 0x0000. When a modulus limit is set, TOF becomes set at the transition from the value set in the modulus register to 0x0000. When the counter is operating in up-/down-counting mode, the TOF flag gets set as the counter changes direction at the transition from the value set in the modulus register and the next lower count value. This corresponds to the end of a PWM period. (The 0x0000 count value corresponds to the center of a period.)

9.5.3   Channel Event Interrupt Description

The meaning of channel interrupts depends on the current mode of the channel (input capture, output compare, edge-aligned PWM, or center-aligned PWM).

When a channel is configured as an input capture channel, the ELSnB:ELSnA control bits select rising edges, falling edges, any edge, or no edge (off) as the edge that triggers an input capture event. When the selected edge is detected, the interrupt flag is set. The flag is cleared by the 2-step sequence described in Section 9.5.1.

When a channel is configured as an output compare channel, the interrupt flag is set each time the main timer counter matches the 16-bit value in the channel value register. The flag is cleared by the 2-step sequence described in Section 9.5.1.

9.5.4   PWM End-of-Duty-Cycle Events

For channels that are configured for PWM operation, there are two possibilities:

- When the channel is configured for edge-aligned PWM, the channel flag is set when the timer counter matches the channel value register that marks the end of the active duty cycle period.
- When the channel is configured for center-aligned PWM, the timer count matches the channel value register twice during each PWM cycle. In this CPWM case, the channel flag is set at the start and at the end of the active duty cycle, which are the times when the timer counter matches the channel value register.

The flag is cleared by the 2-step sequence described in Section 9.5.1.

SECTION 10   OTHER MCU RESOURCES

The microcontroller used in the MPXY8300 Series has several resources which are used to make sensor measurements and/or communicate with the RF transmitter within the device. These modules have a range of programmable features, but are not recommended for user control.

10.1   Analog-to-Digital Converter

The MPXY8300 Series provides a 4-channel, 10-bit analog-to-digital converter (ADC10) module. The ADC10 module is an analog-to-digital converter using a successive approximation register (SAR) architecture with sample and hold.

When making measurements of the various analog voltages the individual blocks need to be powered up long enough to stabilize their outputs before a conversion is started.

All of the features of the MPXY8300 Series ADC10 are the same as similar MC9S08 family devices, except that the analog channels are connected to internal hardware as given in Table 10-1. Conversions are started and ended in the same manner as are the various power up states.

The accuracy, power consumption and timing specifications given in the electrical specifications in Section 17 are based on using the assigned firmware subroutines in Section 14 to make these measurements and convert them into an 8-bit or 9-bit transfer function. These specifications cannot be guaranteed if the user creates custom software routines to make these measurements.

Table 10-1 ADC10 Channel Select

ADCH	ADC10 Channel	Input Select	Firmware Call(s)
00000	AD0	Pressure Sensor	REIMS_READ_COMP_PRESSURE
00001	AD1	Temperature Sensor	REIMS_READ_COMP_TEMP_8
00010	AD2	Bandgap Reference	REIMS_READ_COMP_VOLTAGE
00011	AD3	Reserved	n/a
00100	AD4	Reserved	n/a
00101	AD5	Acceleration Sensor	REIMS_READ_COMP_ACCEL_X REIMS_READ_COMP_ACCEL_Z

10.2   Serial Peripheral Interface

The serial peripheral interface (SPI) on the MPXY8300 Series is only connected and used internally to communicate between the MCU and the RFX. The transfer of data back and forth between the two devices is handled by firmware subroutines as described in Section 14. The user should not attempt to access the RFX using custom software.

10.3   Pressure Measurement

The MPXY8300 Series measures pressure in all members of the family. The pressure measurement consists of a P-Chip Interface (PCI) to a pressure sensing element as shown in Figure 10-1. Control bits on the MCU operate the PCI using firmware provided by Freescale as described in Section 14. The PCI transfers trim data to the P-CHip and measures its resulting analog voltage output. In addition the P-Chip is powered up through the general purpose I/O port.
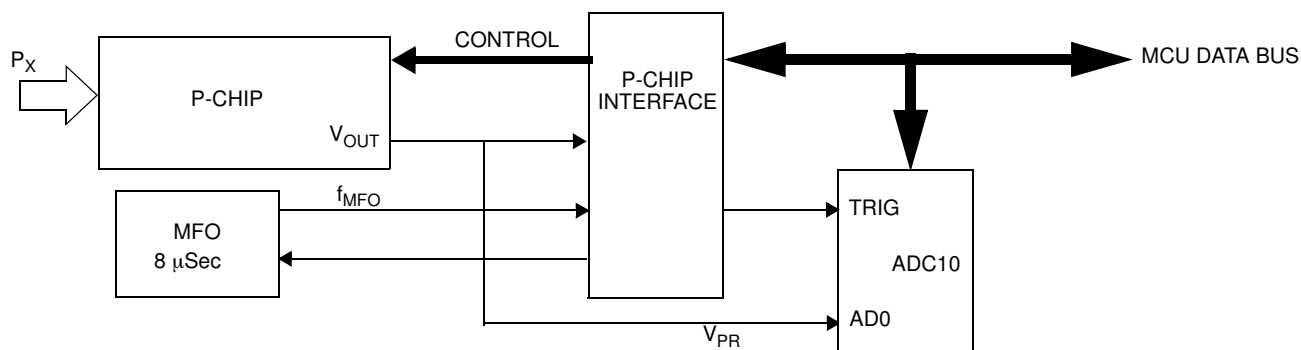
Figure 10-1 PCI to P-Chip Block Diagram

MPXY8300 Series

The signal conditioning of the pressure sensor begins with a capacitance to voltage conversion (C-V) followed by a switched capacitor amplifier. This amplifier has adjustable offset and gain trimming that is set by the trim register. The micro-machined pressure transducer, C-V converter, sample-and-hold, buffer and trim registers are on the separate P-Chip silicon device mounted within the MPXY8300 Series package.

The MCU contains four registers of control, status and trim bits. In addition the P-Chip is powered up through the general purpose I/O driver. The proper sequencing of these bits are handled by the REIMS_READ_COMP_PRESSURE firmware subroutine described in Section 14.

The output values of $000 or $1FF from the calculated pressure measurement are considered fault codes. Further, any calculated value of pressure below minimum calibrated range or above the maximum calibrated range will be converted to the $001 or $1FF codes respectively. The transfer function for pressure versus an 9-bit result from the REIMS_READ_COMP_PRESSURE firmware depends on the calibrated range of the device.

For the 100-450 kPa range the transfer function is:

$$P = 0.686 \times P_{CODE} + 99.314$$

For the 100-800 kPa range the transfer function is:

$$P = 1.375 \times P_{CODE} + 98.625$$

For the 100-1500 kPa range the transfer function is:

$$P = 2.751 \times P_{CODE} + 97.250$$

NOTE

> The accuracy, power consumption and timing specified for a pressure measurement in the electrical specifications in Section 17 are only guaranteed if the user obtains a pressure reading using the firmware subroutine call, REIMS_READ_COMP_PRESSURE.

Faults in the pressure sensor that result in a reading less than the minimum level or greater than the maximum calibrated level will result in the firmware making the output value zero. An output code of zero or 255 are considered fault codes.

Another cross check is that can be performed is that there should be a change in pressure in the presence of a change in temperature. Therefore the user software should cross check the pressure and temperature readings to determine if the device is responding to the Ideal Gas Law:

$$P \times V = n \times R \times T$$

The temperature within a tire will rise after the tire has been in motion for some period of time. This rise in temperature should be accompanied by a rise in pressure. Lack of this pressure rise may be an indication of a plugged pressure port or damaged pressure sensing diaphragm.

NOTE

> The triggering of the ADC10 is shared with the ACI for the acceleration measurements. Both a pressure measurement and acceleration measurement cannot be started before the ADC10 is triggered as there is no status bit to define whether the PCI or the ACI triggered the reading that was captured and converted by the ADC10.

MPXY8300 Series

10.4　Temperature Measurements

The MPXY8300 Series measures temperature in all members of the family. The temperature is measured from a ΔV_B sensor built into channel 1 of the ADC10. Obtaining a temperature measurement is done using the REIMS_READ_COMP_TEMP_8 firmware subroutine as described in Section 14. This subroutine sequences the correct bits to obtain a temperature dependent voltage and convert it to the transfer function specified in Section 17.

The output of this measurement can be used as an over temperature shutdown to prevent RF transmissions and save power during very high temperature conditions which might over stress the battery. This over temperature shutdown software can also set up a restart when the temperature is once again less than the temperature restart level, T_{NORM}, using the temperature restart circuit as described in Section 10.7.2.

The output values of $00 or $FF from the calculated temperature measurement are considered fault codes. Further, any calculated value of temperature below -40°C or above 125°C will be converted to the $00 or $FF fault code respectively. The transfer function for temperature versus an 8-bit result from the REIMS_READ_COMP_TEMP_8 firmware is:

$$T = T_{CODE} - 55$$

NOTE

The accuracy, power consumption and timing specified for a temperature measurement in the electrical specifications in Section 17 are only guaranteed if the user obtains a temperature reading using the firmware subroutine call, REIMS_READ_COMP_TEMP_8.

Faults in the temperature sensor that result in a reading less than the minimum level (-40°C) or greater than the maximum level (125°C) will result in the firmware making the output value zero. An output code of zero or 255 are considered fault codes.

Another cross check is that there should be a change in pressure in the presence of a change in temperature. Therefore the user software should cross check the pressure and temperature readings to determine if the device is responding to the Ideal Gas Law:

$$P \times V = n \times R \times T$$

The temperature within the tire will rise after the tire has been in motion for some period of time. Lack of a temperature rise during the first few minutes of tire motion may be an indication of a failed temperature sensor.

10.5　Voltage Measurements

Voltage measurements can be made on the internal bandgap to estimate the supply voltage on V_{DD}.

10.5.1　Internal Bandgap

An internal bandgap voltage reference is provided to take measurements of a supply voltage. This voltage measurement is taken directly by the ADC10 converter using channel 2 as the bandgap reference is always powered up during any ADC10 conversion.

Obtaining a calculated V_{DD} voltage measurement based on the bandgap reference is done using the REIMS_READ_COMP_VOLTAGE firmware subroutine as described in Section 14. This subroutine sequences the correct bits to obtain a voltage measurement and convert it to the transfer function specified in Section 17.

The output values of $00 or $FF from the calculated voltage measurement are considered fault codes. Further, any calculated value of voltage below 2.1V or above 3.6V will be converted to the $00 or $FF fault code respectively. The transfer function for voltage versus an 8-bit result from the REIMS_READ_COMP_VOLTAGE firmware is:

$$T = 0.01 \times T_{CODE} + 1.22$$

NOTE

The accuracy, power consumption and timing specified for a voltage measurement in the electrical specifications in Section 17 are only guaranteed if the user obtains a voltage reading using the firmware subroutine call, REIMS_READ_COMP_VOLTAGE.

MPXY8300 Series

10.6   Acceleration Measurements

The acceleration measurement consists of an analog interface (ACI) to a acceleration sensing element as shown in Figure 10-2. Control bits on the MCU operate the ACI to power up the g-Cell and capture a voltage which is converted by the ADC10.

The signal conditioning of the g-Cell begins with a capacitance to voltage conversion (C-V) followed by a switched capacitor amplifier. This amplifier has adjustable offset and gain trimming that is set by the trim register. The micro-machined pressure transducer, C-V converter, sample-and-hold, buffer and trim registers are on the MCU and the g-Cell is a separate silicon device mounted within the same package as the P-Chip, RFX and MCU.

The MCU contains four registers of control, status and trim bits. The proper sequencing of these bits are handled by the REIMS_READ_COMP_ACCEL_X/Z firmware subroutine described in Section 14.

In order to trim the acceleration response of each device the MCU will transfer the trim data to the trim registers in the ACI from parameters stored in a designated area of the FLASH memory during the manufacturing process. These trim registers remain powered up as long as the ACI is powered up, but can also be updated at any time dependent on the user software by calling the REIMS_ACITD subroutine in the firmware as described in Section 14. As an aid to determine if the trim data has been corrupted in the application (due to a severe EMI event) a parity check circuit is always checking the trim register. The normal case is for the parity of all trim bits (including the added parity bit) is to be an odd number. If the parity ever switches to an even number of bits then the V_{ACC} output from the ACI will be forced to V_{MIN} which is near V_{SS}, which yields a result near \$00 from the ADC10. If this occurs the user software should initiate a trim update and check that the parity problem has been solved.
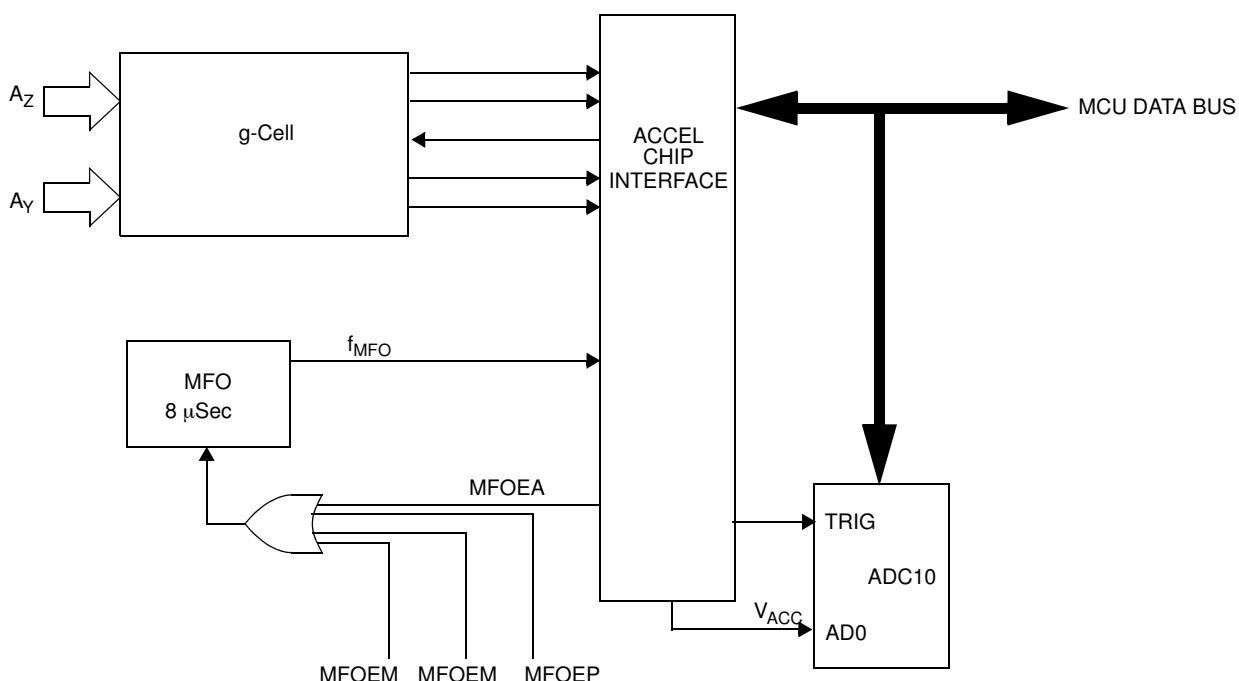
Figure 10-2 ACI to g-Cell Block Diagram

The g-Cell contains two acceleration sensing elements with one oriented to sense acceleration along the axis through the top of the package (Z-axis) and along the axis that is parallel to the rows of pins on the package (X-axis). The device needs to be oriented in the application to obtain the desired acceleration axis.

There is only one acceleration processing signal chain and each axis must be measured as a separate measurement. The ZSEL bit in the ACIC register selects the Z-axis when it is set and the X-axis when it is clear. The polarity of the Z-axis signal can be reversed from that shown in Figure 2-2 by a bit in the trim data. User must specify the desired polarity for the final manufacturing trim/test process. The response of either axis of the g-Cell can be filtered with a simple 2-pole, 500 Hz low pass filter. These control bits can be controlled by the REIMS_READ_COMP_ACCEL_X/Z firmware routine as passed parameters.

The output values of \$00 or \$FF from the calculated acceleration measurement are considered fault codes. Further, any calculated value of acceleration outside the calibrated range of operation will be converted to the \$00 or \$FF fault code. The transfer function for X-axis acceleration versus an 9-bit result from the REIMS_READ_COMP_ACCEL_X firmware is:

$$A \ = \ 0.039 \times A_{CODE} - 10.039$$

The transfer function for Z-axis acceleration versus an 9-bit result from the REIMS_READ_COMP_ACCEL_Z firmware is:

$$A = 0.118 \times A_{CODE} - 0.118$$

NOTE

The accuracy, power consumption and timing specified for an acceleration measurement in the electrical specifications in Section 17 are only guaranteed if the user obtains an acceleration reading using the firmware subroutine call, REIMS_READ_COMP_ACCEL_X/Z which uses the 500 Hz low-pass filter.

NOTE

The triggering of the ADC10 is shared with the PCI for the pressure measurements. Both a pressure measurement and acceleration measurement cannot be started before the ADC10 is triggered as there is no status bit to define whether the PCI or the ACI triggered the reading that was captured and converted by the ADC10.

NOTE

The trim data in the ACITM0:3 registers (accessed using the ATD0:7, ATMRW, and the ATMSEL0:1 bits) should not be altered directly with the MCU. These bits should only be updated from FLASH memory using the REIMS_ACITD firmware routine.

The accelerometers can be cross checked against each other to determine any permanent faults. The X-axis accelerometer should be cycling ±1g at a rate up to 50 Hz, if the Z-axis accelerometer is indicating any continuous acceleration over 5g. Similarly, the Z-axis accelerometer should indicate less than 5g if the X-axis accelerometer indicates no cycling greater than 1g.

10.7   Thermal Shutdown

When the package temperature becomes too low or too high the MCU can go into a stop mode to suspend operation and prevent transmission of RF signals which may be corrupted at the temperature extremes.

10.7.1   Low Temperature Shutdown

Low temperature shutdown is achieved using temperature readings taken by the ADC10 as described in Section 10.1. When the software programmed low temperature is reached the MCU will set a bit in the parameter registers, turn off the RFX and enter the Stop1 mode. The PWU will restart the MCU which will check the flag bit first. If set the MCU will first check the temperature to determine if the shutdown should continue.

10.7.2   High Temperature Shutdown

The high temperature shutdown level is determined from a measurement of the temperature sensor by the ADC10 as described in Section 10.1.

Normally the user software will determine when to shutdown and then turn off all power and place the MCU in the stop1 mode. The MCU, temperature sensor, ADC10 and PWU are all functional over the full temperature range from T_L to T_H.

The MCU can be restarted by the Temperature Restart (TR) module when the temperature falls back below a lower temperature level, T_{RESET}. When this occurs the MCU will be reset and begin execution from the reset vector located at $DFFE/$DFFF. The TR module cannot activate an MCU reset unless it has first risen above a higher temperature, T_{REARM}, as shown in Figure 10-3. Both T_{RESERT} and T_{REARM} as the limits of the nominal detection level of T_{NORM}.

The status of the TR can be checked by reading the TRO bit located at bit 0 in the SIMTST register at address $180F. The TRO bit is set high by an MCU reset. The state of the TRO bit is as follows:

$1 =$ TR module has reached the T_{REARM} temperature and will restart the MCU if the temperature falls back below the T_{RESET} temperature.

$0 =$ TR module has been below the T_{RESET} temperature and cannot restart the MCU. The MCU will <u>NOT</u> go into either the stop1 or stop2 mode.
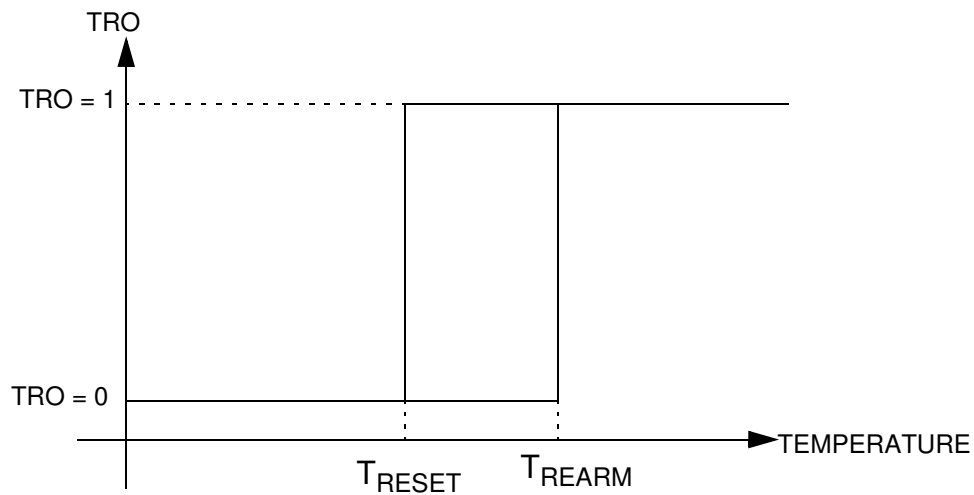
Figure 10-3 Temperature Restart Response

The TR module can be powered on and off by setting or clearing the TRE bit located at bit 7 in the SOPT2 register at address $1803. The TRE bit is cleared by an MCU reset.

NOTE

It is recommended that the wake-up time be disabled after setting the periodic reset time to it maximum value in order to prevent an unwanted wake-up of the MCU before the device has had a chance to cool down. See Section 11 for details on the control bits for the periodic wake-up module.

SECTION 11 PERIODIC WAKE-UP TIMER

The periodic wake-up timer (PWU) generates a periodic interrupt to wake-up the MCU from any of the low power modes. It also has an optional periodic reset to restart the MCU. It is driven by the LFO oscillator in the RTI module which generates a clock at a nominal one millisecond interval. The LFO and the wake-up timer are always active and cannot be powered off by any software control. The control bits are set so that there is either a periodic wake-up, a periodic reset, or both a wake-up interrupt and a periodic reset. No combination of control bits will disable both the wake-up interrupt and the periodic reset. In addition, there is no hardware control that can mask a wake-up interrupt once it is generated by the PWU.

11.1 Block Diagram

The block diagram of the wake-up timer is shown in Figure 11-1. This consists of a programmable pre-scaler with 64 steps that can be used to adjust for variations in the value of the LFO period. Finally there are two cascaded programmable 6-bit dividers to set wake-up and/or reset time intervals.
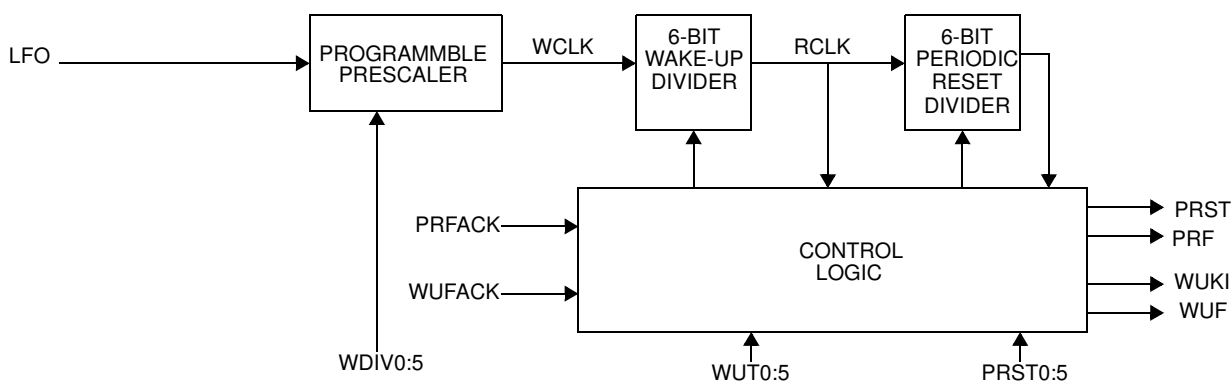
Figure 11-1 Wake-Up Timer Block Diagram

The wake-up divider (PWUDIV) register selects a division of the incoming 1 msec clock to generate a wake-up clock, WCLK. The WCLK frequency can be calibrated against the more precise external oscillator using the TCAL firmware subroutine as described in Section 14. This subroutine turns on the RFX crystal oscillator and feeds a 500 kHz clock to the TPM1 via the internal Dx signal for one cycle of the LFO. The measured time is used to calculate the correct value for the WDIV0:5 bits for a WCLK period of 1 second. The TCAL subroutine cannot be used while the RFX is transmitting or the TPM1 is being used for another task.

The wake-up time register (PWUSC0) selects the number of WCLK pulses that are needed to generate a wake-up interrupt to the MCU. The periodic reset register (PWUSC1) selects the number of wake-up pulses that are needed to generate a periodic reset of the MCU. If both the reset and the interrupt occur on the same clock cycle the reset will have precedence and the interrupt will not be generated.

NOTE

The wake-up interrupt (WUKI) cannot be masked by any hardware control. Therefore clearing the I-bit will not prevent a wake-up interrupt.

11.2 Wake-Up Divider Register - PWUDIV

The PWUDIV register contains six bits to select the division of the incoming 1 msec clock period as described in Figure 11-2.

$0038

	Bit 7	6	5	4	3	2	1	Bit 0
R	0	0			WDIV			
W								
RESET:	—	—	—	—	—	—	—	—
POR:	0	0	0	1	1	1	1	1

Figure 11-2 PWU Divider Register (PWUDIV)

Table 11-1 LFCS1 Register Field Descriptions

Field	Description
7:0 unused	Unused register bits, always read as 0.
5:0 WDIV	**Wakeup Divider** — These bits select an incoming pre-scaler for the incoming 1 msec clock period from 504 to 1512. This results in a clocking of the 6-bit wake-up divider at rates from a nominal 504 to 1512 msec for each wake-up clock, WCLK. The user can use this prescaler to fine tune the wake-up time based on the variation in the LFO frequency. The conversion from the decimal value of the WDIV bits to the nominal WCLK period is given as: $$t_{WCLK} = \frac{(504 + 16 \times WDIV)}{1000}$$ A power on reset presets these bits to a value of $1F (decimal 31) which yields a nominal 1 second output period for WCLK. Other resets have no effect on these bits.

11.3 PWU Control/Status Register 0 - PWUCS0

The PWUCS0 register contains six bits to select the division of the incoming WCLK clock period and provide interrupt flag and acknowledge bits as described in Figure 11-3. The period of the resulting interrupt also generates the clock, RCLK, for the periodic reset timing.

$0039

	Bit 7	6	5	4	3	2	1	Bit 0
R	WUF	0			WUT5			
W		WUFACK						
RESET:	0	—	1	1	1	1	1	1

Figure 11-3 PWU Control/Status Register 0 (PWUCS0)

Table 11-2 PWUCS0 Register Field Descriptions

Field	Description
7 WUF	**Wake-Up Interrupt Flag** — The WUF bit indicates when a wake-up interrupt has been generated by the PWU. This bit is cleared by writing a one to the WUFACK bit. Writing a zero to this bit has no effect. Reset clears this bit. 0 Wake-up interrupt not generated or was previously acknowledged 1 Wake-up interrupt generated
6 WUFACK	**Wake-Up Interrupt Flag Acknowledge** — The WUFACK bit clears the WUF bit if written with a one. Writing a zero to the WUFACK bit has no effect on the WUF bit. Reading the WUFACK bit returns a zero. Reset has no effect on this bit. 0 No effect 1 Clear WUF bit
5:0 WUT	**Wake-Up Time Interval** — These control bits select the number of WCLK clocks that are needed before the next wake-up interrupt is generated. The count gives a range of wake-up times from 1 to 63 WCLK clocks. Depending on the value of the bits for the WDIV0:5 this time interval can nominally be from 1 to 63 seconds in 1 second steps. Whenever the WUT0:5 bits are changed the prescaler, wake-up interrupt counter and periodic reset counters are restarted. Writing the same data to the WUT0:5 bits has no effect. Writing zeros to all of the WUT0:5 bits forces the wake-up divider to a value of $3F and disables the wake-up interrupt. However, writing all zeros to the WUT0:5 bits is inhibited if all of the PRST0:5 bits are already cleared to zero. This prevents disabling both the periodic wake-up and the periodic reset at the same time as shown in Table 11-3. The WUT0:5 bits are preset to a value of $3F (decimal 63) by any resets.

Table 11-3 Limitations on Clearing WUT/PRST

Control Bits	State of Control Bits	Control Bits to be Cleared	Resulting Action	Resulting Wake-Up Interrupt	Resulting Periodic Reset
WUT0:5	non-zero	PRST0:5	Allowed	Enabled[1]	Disabled
	all zero		Inhibited	Disabled[2]	Enabled[1]
PRST0:5	non-zero	WUT0:5	Allowed	Disabled	Enabled[1]
	all zero		Inhibited	Enabled[1]	Disabled

NOTES:
    1. Using previous values.
    2. Wake-up divider preset to $3F.

11.4   PWU Control/Status Register 1 - PWUCS1

The PWUSC1 register contains six bits to select the division of the incoming RCLK clock period and provide interrupt flag and acknowledge bits as described in Figure 11-4.

$003A

	Bit 7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit 0**
R	PRF	0			PRST			
W		PRFACK						
RESET:	0	0	1	1	1	1	1	1

Figure 11-4 PWU Control/Status Register 1 (PWUCS1)

Table 11-4 PWUCS1 Register Field Descriptions

Field	Description
7 PRF	**Periodic Reset Flag** — The PRF bit indicates when a periodic reset has been generated by the PWU. MCU writes to this bit have no effect. This bit is cleared by writing a one to the PRFACK bit. This bit is cleared by a power on reset, but is unaffected by other resets. Periodic reset not generated or previously acknowledged Periodic reset generated
6 PRFACK	**Periodic Reset Flag Acknowledge** — The PRFACK bit clears the PRF bit if written with a one. Writing a zero to the PRFACK bit has no effect on the PRF bit. Reading the PRFACK bit returns a zero. Reset has no effect on this bit. No effect Clear PRF bit
5:0 PRST	**Periodic Reset Time Interval** — These control bits select the number of wake-up interrupts that are needed before the next periodic reset is generated. The decimal count gives a range of periodic reset times from 1 to 63 wake-up interrupts. Depending on the value of the bits for the WDIV0:5 and WUT0:5 this time interval can nominally be from 1 second to 66 minutes with steps from 1 to 63 seconds. Whenever the PRST0:5 bits are changed the prescaler, wake-up interrupt counter and periodic reset counters are restarted. Writing the same data to the PRST0:5 bits has no effect. Writing zeros to all of the PRST0:5 bits forces the periodic reset to be disabled if at least one of the WUT0:5 bits is set to a one. This assures that there will be at least a wake-up interrupt. However, writing all zeros to the PRST0:5 bits is inhibited if all of the WUT0:5 bits are already cleared to zero. This prevents disabling both the periodic wake-up and the periodic reset at the same time. See Table 11-3. The PRST0:5 bits are preset to a value of 63 by any resets.

11.5 Functional Modes

PWU module will work in each of the MCU operating modes as follows:

11.5.1 RUN/WAIT Mode

If the module generates a wake-up interrupt the PC (Program Counter) will be redirected to the wake-up timer interrupt vector. The WUF flag will be set to indicate wake-up timer interrupt; write 1 to WUFACK to clear this flag.

If the module generates a periodic reset the PC will be redirected to the reset vector. The PRF flag will be set to indicate periodic reset; write 1 to PRFACK to clear this flag.

All registers will continue to hold their programmed values after interrupt or reset is taken.

11.5.2 STOP4/STOP3 Mode

If the module generates a wake-up interrupt the bus and core clocks will be restarted and the PC will be redirected to the wake-up timer interrupt vector. The WUF flag will be set to indicate wake-up timer interrupt, write 1 to WUFACK to clear this flag.

If the module generates a periodic reset the bus and core clocks will be restarted and the PC will be redirected to the reset vector. The PRF flag will be set to indicate periodic reset; write 1 to PRFACK to clear this flag.

All registers will continue to hold their programmed values after interrupt or reset is taken.

11.5.3 STOP2/STOP1 Mode

If the module generates a wake-up interrupt the module will cause the MCU to exit the power saving mode as a POR. MCU will have the wake-up interrupt pending and once CLI opcode is executed PC will be redirected to wake-up interrupt vector address. The WUF flag will be set to indicate wake-up timer interrupt, write 1 to WUFACK to clear this flag.

If the module generates a periodic reset the module will cause the MCU to exit the power saving mode as a POR. The PRF flag will be set to indicate periodic reset; write 1 to PRFACK to clear this flag. The SIMRS register will have just the POR bit set.

In this stop mode exit all registers will continue to hold their programmed values.

11.5.4 Active BDM/Foreground Commands

The PWU is frozen when in the Active Background Mode or executing foreground commands, so PWU counters will also be stopped. Normal PWU operation will resume as MCU exits BDM or the foreground command is finished.

SECTION 12 LF RECEIVER

This module provides an interface between magnetic field communication transmissions (typically at 125 kHz) and the MCU. Two pins are connected to a receiver coil located on the printed circuit board. When a magnetic field is present, low level voltages are applied to the input pins. The module amplifies the signal and determines whether the proper carrier frequency is present. If valid information is detected, the module will provide an interrupt request to the CPU. If no signal is detected or the information is invalid, the module will return to a very low power state and periodically wake-up and sample the input.

The module may be placed in three modes:

Carrier detect mode — This mode determines whether the received signal is above a certain amplitude, is present for a minimum duration, and has a carrier frequency between a minimum and maximum level. If the three criteria are met, the CPU receives an interrupt request.

Manchester data mode — This mode is for receiving Manchester coded information. The information being transmitted will be modulated using on/off keying and will have a carrier frequency of approximately 125 kHz. The module will use automatic gain control to acquire and track the incoming signal, which may vary greatly due to changes in antenna distance and orientation from the transmitter. The module will monitor: amplitude, carrier frequency, decoded bit timing, and message ID. If all of the criteria are met, the CPU will be interrupted when each byte of information is available.

Direct access mode — This mode is provided primarily for system development when the antenna characteristics are being analyzed. In this mode, the CPU is directly provided the demodulated data stream. As in Manchester data mode, automatic gain control will be used to acquire and track the input waveform.

The module provides a programmable periodic wake-up mechanism for carrier and Manchester data modes. This allows the module to be normally in a very low power state and to periodically wake-up to sample for incoming information. In direct access mode, the module will stay active. The block diagram of the module is shown in Figure 12-1. The module is divided into two sub-blocks:

Detector — Analog and digital sub-block that performs:
- Signal amplification
- On/off keying demodulation
- Carrier frequency digital bandpass filter
- Automatic gain control

Decoder — Digital logic that performs:
- Interface to the CPU bus
- Manchester bit timing decoding
- Wake-Up timing

12.1 External Signal Description

12.1.1 Modulated Signal input

Two port pins act as inputs to the LFR detector.

12.1.2 Clock inputs

A medium frequency oscillator (MFO) of approximately 125 kHz will be provided to the module to clock the logic of the decoder. A frequency doubled version of the MFO is supplied to the detector. The MFO will have an enable signal from the LFR module.

A low frequency oscillator (LFO) of approximately 1 kHz will be provided to the module at all times in all MCU modes of operation.
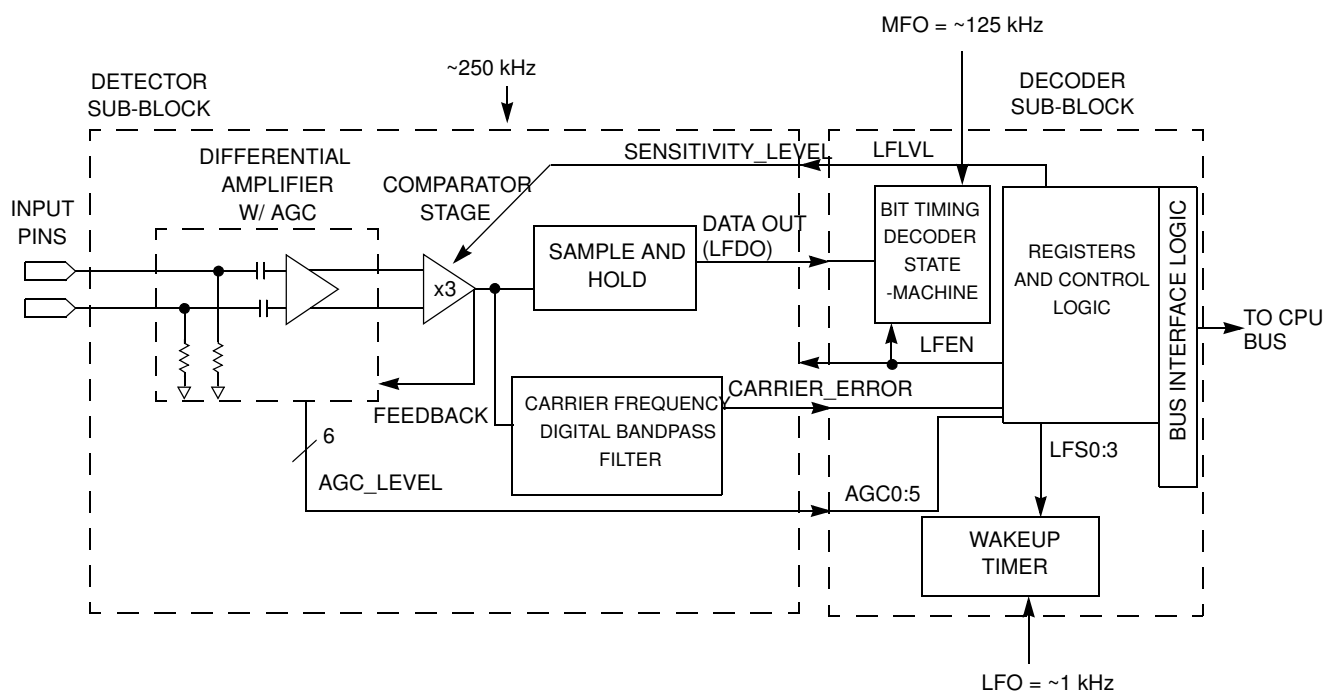
Figure 12-1 LF Receiver with Automatic Gain Control Block Diagram

12.2 Functional Description

12.2.1 Write to the LFR Registers

The user can always write to the LFEN, LFAGC, LFDFAK, LFCFAK and LFIG bits regardless of the status of LFEN.

A write to registers: LFCA0, LFCA1, LFCB0, LFCB1, LFCR and LFCS0 (LFIE, LFRST, LFWU8 and LFLVL bits) is effective only if LFEN is zero. It is possible to write to LFCS0 in one instruction to both set LFEN and set or clear the other control bits in that register.

12.2.2 LFR Modes of Operation

The LFM1 and LFM0 bits define the LFR mode of operation:

- MCU Direct
- Carrier Detect
- Manchester Data

The LFR can be active in any of the low power modes: stop1, stop2, stop3, and stop4

The function of the LFR module is summarized in Table 12-1.

Table 12-1 LF Decoder Modes

Decoding Mode	Data Received	LFR Sensitivity	LF AGC Enabled	MCU Control	Detector Sampling Rate	LFR Detector On Time	
						LFON = 1	LFON = 0
MCU Direct	Determined by MCU software monitoring LFDO	Selected by LFLVL	Yes	Full control	Continuous on or off selected by LFEN	N/A	
Carrier Detect	No		No	Interrupt driven	Selected by LFS[3:0]	265 (256 + 9) MFO periods (approximately 2120 μs)	25 (16+9) MFO periods (approximately 200 μs)
Manchester Data	Yes		Yes			With no data detected approximately 200 μs	

12.3   MCU Direct Mode

In the MCU direct mode, the LFR detector block is enabled and user software can directly monitor the data coming from the detector (LFDO status bit). The decoder is bypassed. User software can control the LFR detector using the LFEN bit. The level of sensitivity can be adjusted with the LFLVL bit. The automatic gain control function of the detector block will be enabled. User software should allow time, t_{ACQ} before monitoring LFDO. The bandpass pass filter will be enabled and can be monitored by reading the LFBF bit.

In MCU direct mode, the LFR decoder turns on the MFO in the MCU bus clock domain when LFEN is set. After two rising edges of the MFO clock the LFR detector is turned on.

In the MCU direct mode the auto-zero function in the amplifier will only be activated when:

    •   the LFR detector is powered up from an off state

    •   when the LFDO output of the LFR detector goes from a one to zero.

This auto-zero function takes 4 MFO periods (approximately 32 μsec) to complete. At high temperatures (>100°C) it will be necessary for user software to reset the module to execute an auto-zero operation when there is no activity being detected.

The LFS[3:0] bits are ignored in the MCU direct mode.

12.4   Carrier Detect Mode

In the carrier detect mode the LFR detector is periodically powered up to sample the signals applied to the LFR detector.

In this mode the MFO and LFR detector remain turned on during the t_{LFON} time. In this case the value of the t_{LFON} time is defined by the LFON bit. The LFR decoder turns on the LFR detector and MFO on a rising edge of the LFO clock as shown in Figure 12-2. The first nine MFO cycles are used to make the LFR initialization and includes the start-up time of the MFO, the time the LFR detector needs to make an auto-zero and the time that the LFR decoder needs to start sampling.

The LFR detector remains turned on for a period of t_{LFON} - 2 cycles of the MFO. In this case, the value of t_{LFON} time is defined by the LFON bit. Following the detector on time the detector remains off or reset for a time period, t_{RST}, set by the LFRST and LFS[3:0] control bits.

The periodic rate of the sampling, t_{SAMPLE}, is defined as:

$$t_{SAMPLE} = t_{LFON} + t_{RST}$$

Examples of the t_{LFON}, t_{RST} and t_{SAMPLE} timing are shown in Figure 12-3 and Figure 12-4 for the two cases of the LFON bit.
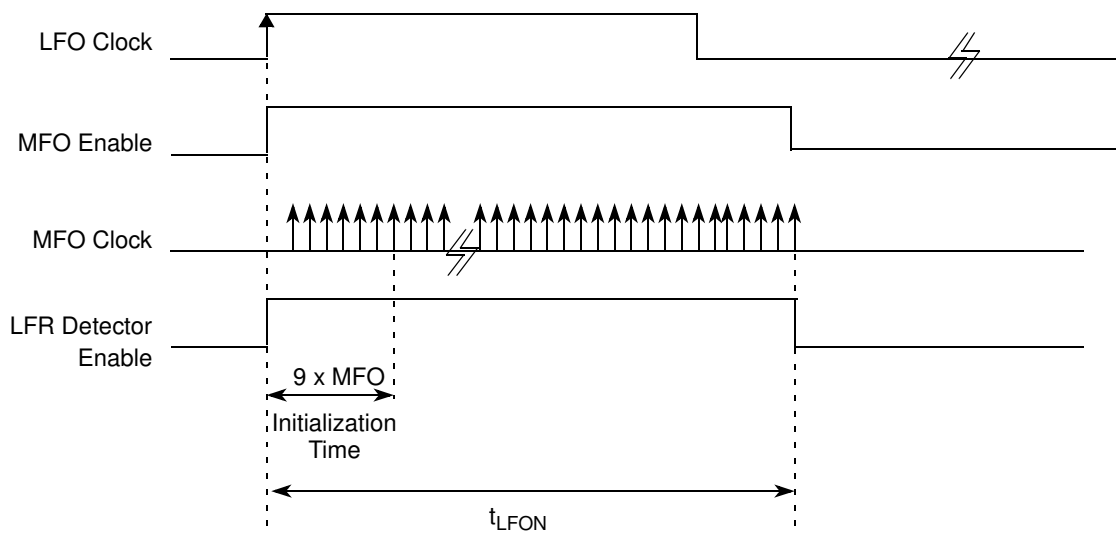

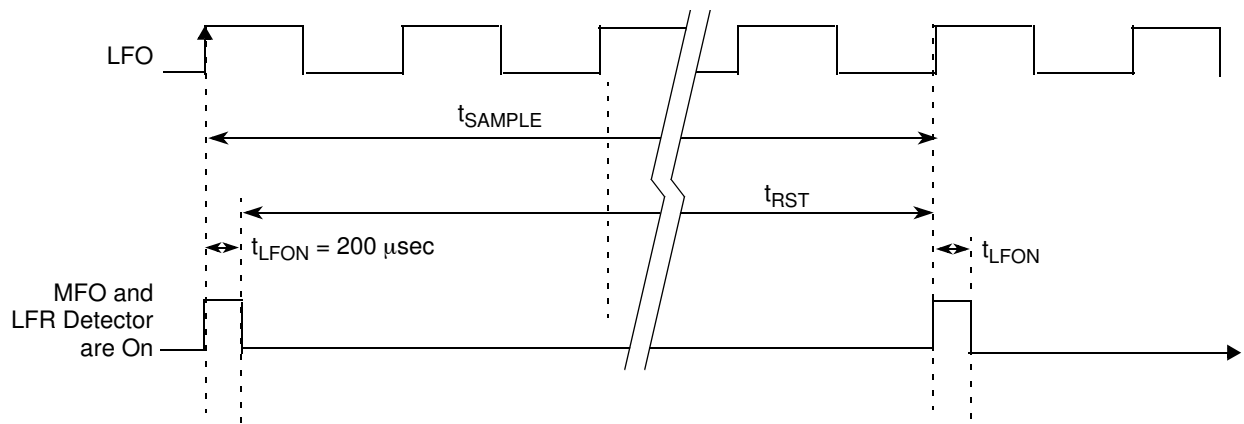**Figure 12-2 LFR Initialization in Carrier Detect Mode**

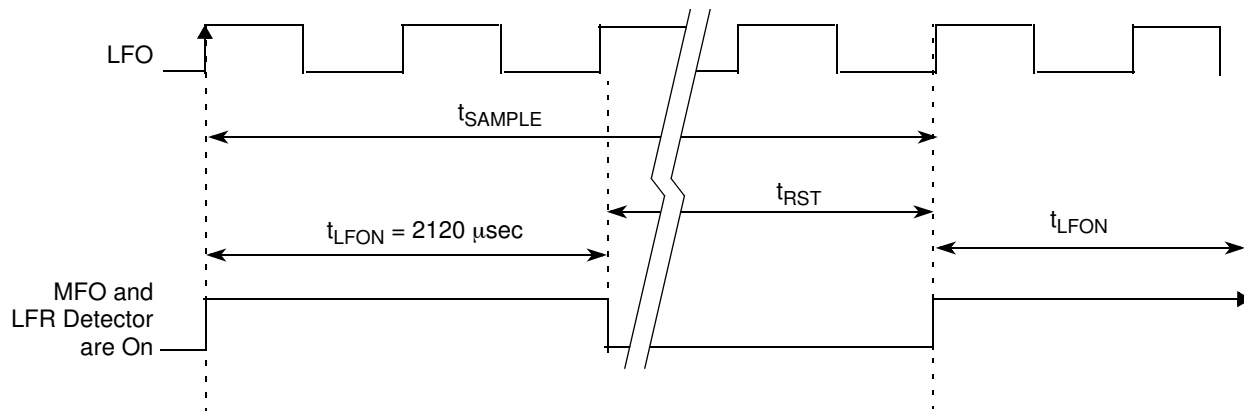**Figure 12-3 LFR Sample Timing - Carrier Detect Mode (LFON = 0 and no signal detected)**

Figure 12-4 LFR Sample Timing - Carrier Detect Mode (LFON = 1 and no signal detected)

In the carrier detect mode the LFR decoder looks for a minimum carrier signal time, t_{CD}, for various cases of LFR detector sampling are shown in Figure 12-5, Figure 12-6, and Figure 12-7. The carrier detection time, t_{CD}, is defined by the LFCT bit.
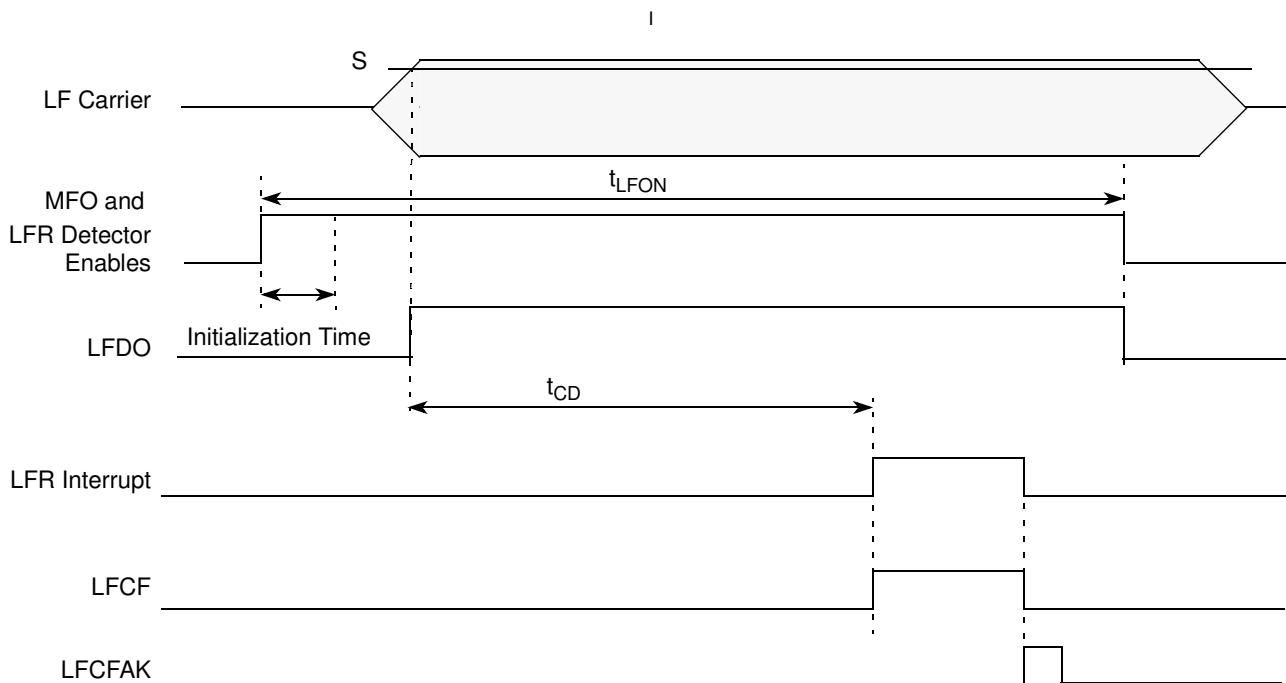
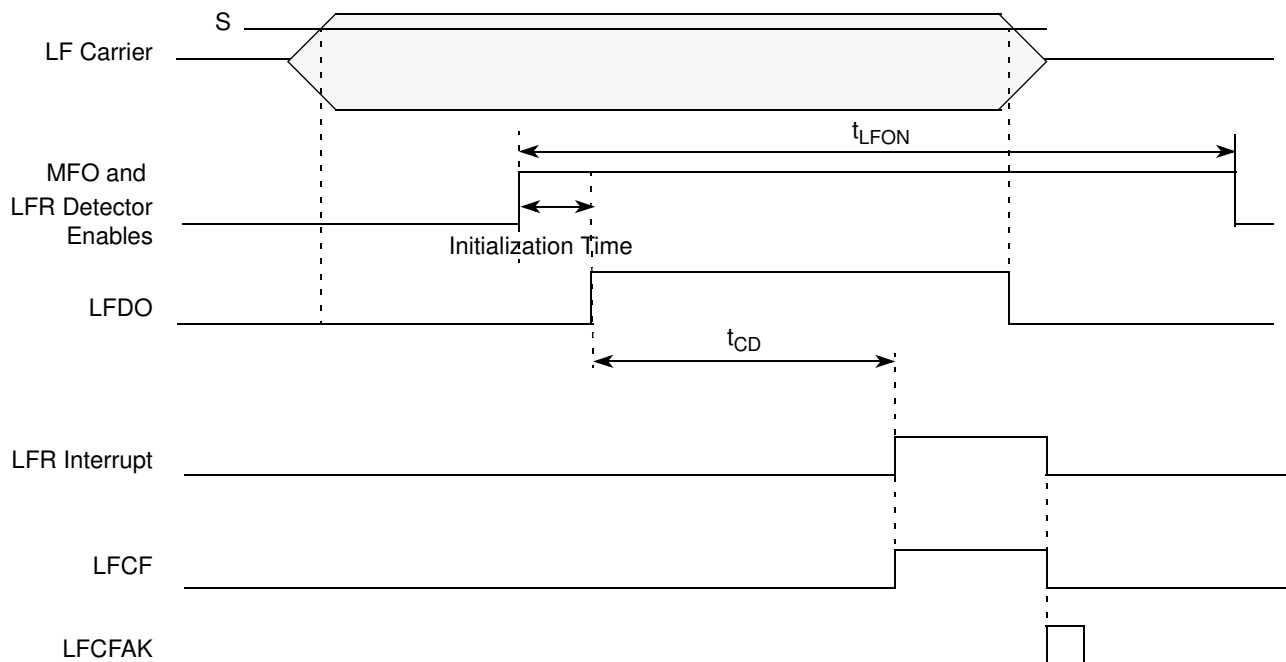**Figure 12-5 Carrier Detect Mode — LFR Detector On Before Carrier**

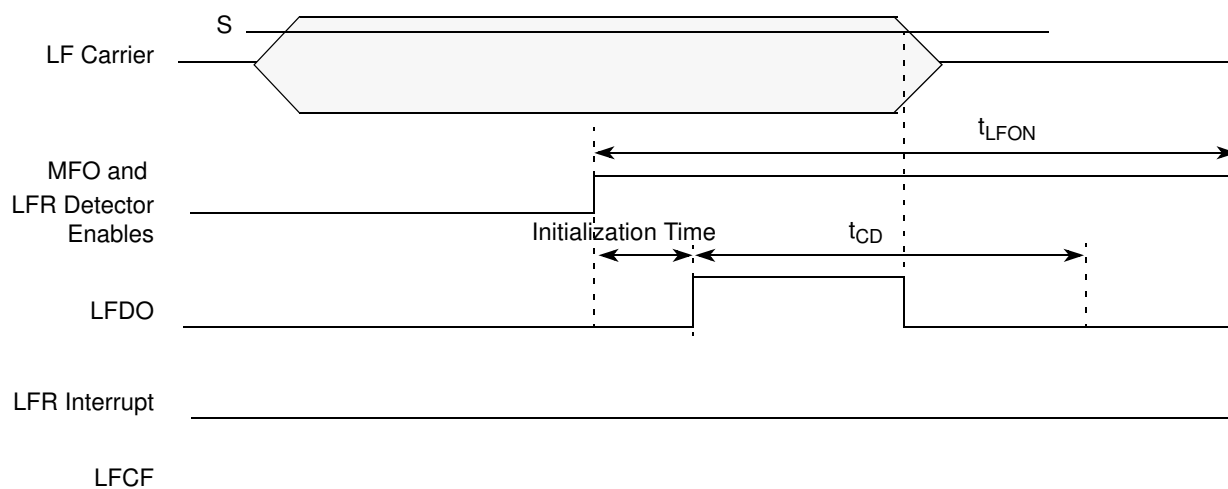**Figure 12-6 Carrier Detect Mode — LFR Detector On After Carrier**

Figure 12-7 LF Carrier Detect Mode — LF Detector On Too Late

The sensitivity level, S, is selected by the LFLVL bit.

After the carrier detection time, t_{CD}, is reached the LFR decoder sets the LFCF bit and LFR interrupt to the MCU (if the LFIE bit is set). The LFR interrupt and the LFCF bit are cleared when a logical one is written to the LFCFAK bit.

When the t_{LFON} period ends, the LFR decoder turns off the LFR detector and MFO, regardless of the length of time that the carrier exists.

12.5 Manchester Data Mode

In the Manchester data mode the incoming data stream has each data bit defined by a time interval with either a rising or falling edge in the middle of the time frame as shown in Figure 12-8. The rising and falling edge are determined by a signal level, S. The nominal data rate is 4 kbits/sec which makes each data bit time, t_{DATA}, approximately 250 μsec with a pulse width of 125 μsec. Data rates covering the complete range of Dr(min) to Dr(max) kbits/sec must be decoded as valid Manchester data. Data rates below F_{DATA_L} and above F_{DATA_H} will always be rejected.

A logical one is defined as no LF carrier present for the first half of the bit time; and a logical zero is defined as a LF carrier present for the first half of the bit time as shown in Figure 12-8.
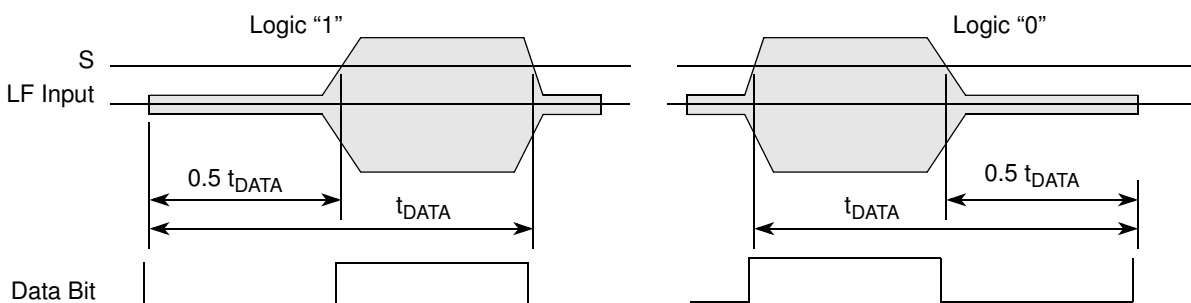

Figure 12-8 Manchester Bit

The format of the complete Manchester datagram is shown in Figure 12-9. It is comprised of a preamble, a synchronization period, a wake-up code, and at least one data byte.
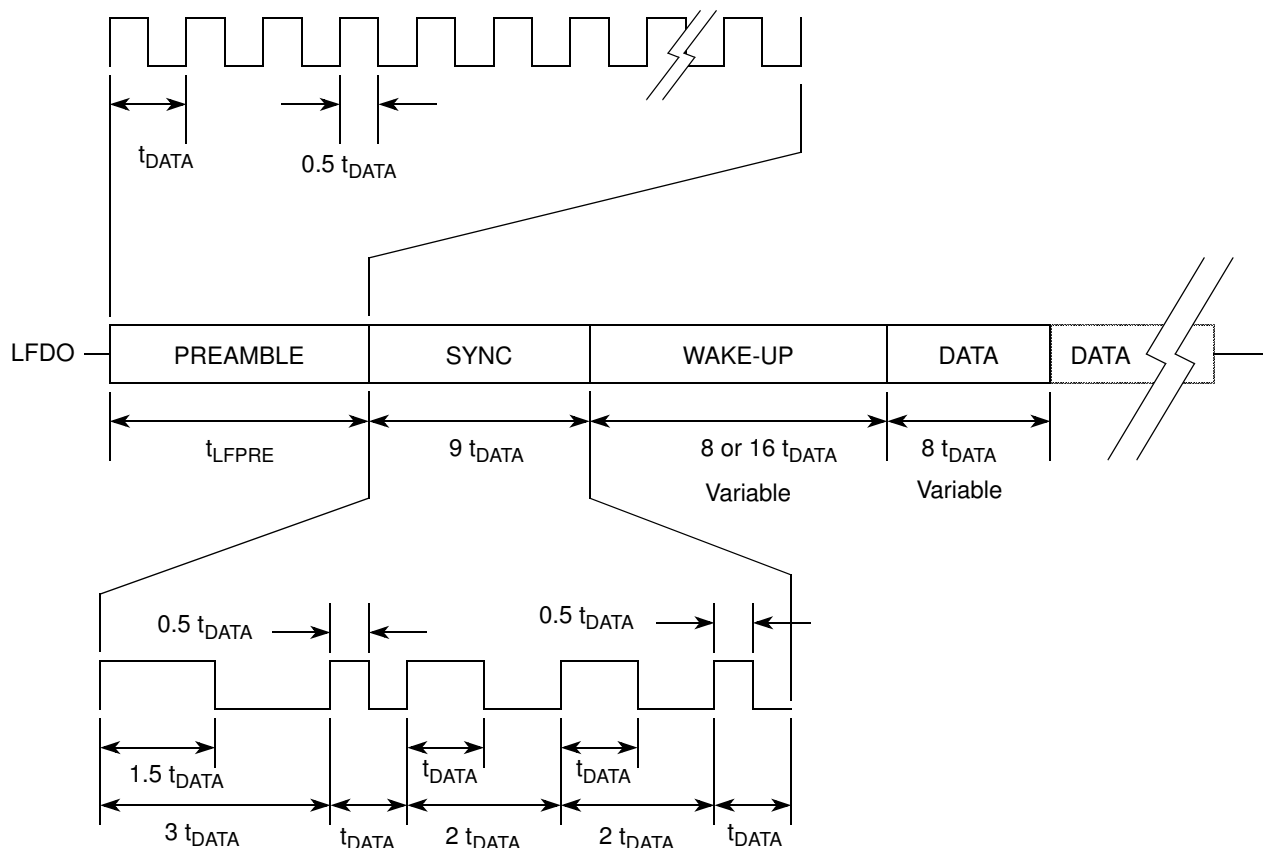
Figure 12-9 Manchester Datagram Format

12.5.1   Sample Time in Manchester Data Mode

In the Manchester data mode the LFR detector is powered up periodically, t_{SAMPLE}, to sample for signals applied to the LFR detector. The periodic rate of the sampling, t_{SAMPLE}, is defined as:

$$t_{SAMPLE} = t_{LFON} + t_{RST}$$

In the beginning of t_{SAMPLE} time, LFR decoder turns on the MFO and the LFR detector at the rising edge of the LFO. The LFR detector and MFO remain turned on during t_{LFON} time.

If LFR detector does not receive a signal greater than the detect threshold during t_{LFON} time, the LFR decoder turns off the LFR detector and the MFO as shown in the example in Figure 12-10.

During the t_{LFON} time if a signal greater than the detect threshold is received, the LFR detector attempts to acquire the signal with the automatic gain control (AGC). The LFR detector needs to receive 5 Manchester zeros to acquire the signal and set the AGC level. This time is referred to as t_{ACQ}.

If an error is detected, the module will be placed in a reset state for the duration set by LFS[3:0]. An error may be in the form of:

1. Carrier frequency out of bounds
2. Data rate out of bounds
3. No signal present
4. 2 msec period with no signal acquired

An interrupt signal is not generated for an error. See Figure 12-11 for an example of an error sequence.

If a signal is present during t_{LFON} time, but ends before being acquired, then the LFR decoder will turn off the LFR detector after a maximum of two LFO cycles.
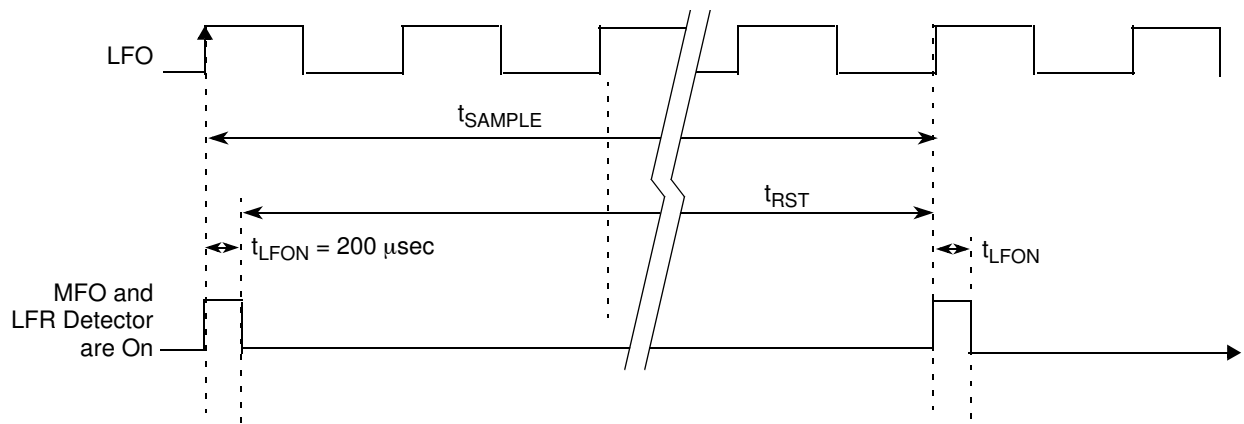
Figure 12-10 LFR Sample Timing - Manchester Data Mode (no signal detected during t_{LFON})
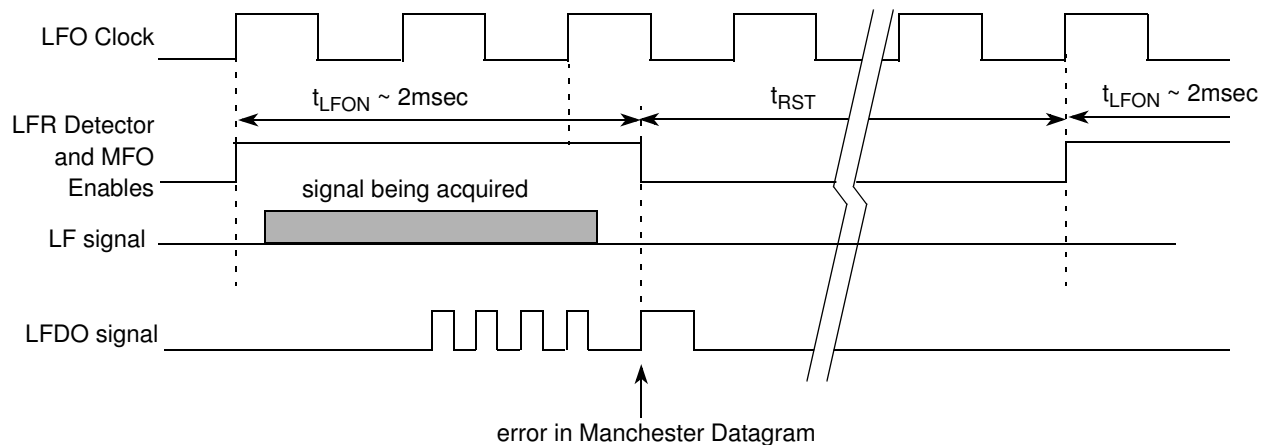

Figure 12-11 LFR Sample Timing - Manchester Data Mode - data error detected

12.5.2   Preamble

The preamble consists of a sequence of Manchester zeros to signal the start of valid information. This preamble must last a minimum of the 8 Manchester zeros. It is acceptable if the preamble has more than 8 Manchester zeros.

The preamble ends when the incoming signal active pulse width lasts for 1.5 Manchester data bit times. If the number of the received Manchester zeros during the preamble (until the 1.5 data sync pulse) is less than t_{PRE}(min), then the incoming signals are ignored, the LFR detector and MFO are turned off and the decoder is reset for a period of time, t_{RST}.

If the correct bits and preamble time are received the LFR detector and the MFO continue to remain powered up to allow the synchronization period to be decoded.

The LFR decoder has a time-out mechanism that turns off the LFR detector and MFO for t_{RST} when the LFR decoder receives only Manchester zeros (in preamble) for an extended time, $t_{TIMEOUT}$, which is defined by counting 65 rising edges of the LFO clock. The time-out counter will always be enabled when the module is first enabled and will begin after the signal has been acquired and valid Manchester data coded zeros are received.

12.5.3   Synchronization

In order for the LFR decoder to synchronize to the upcoming wake-up code, a specific pattern of encoded data must be received during the synchronization interval which lasts a total of nine (9) bit times. If the synchronization pattern fails to match the correct one, the LFR detector and the MFO are turned off and the LFR decoder is reset for t_{RST}.

If the correct synchronization is decoded the LFR detector and the MFO remain powered up to allow the wake-up code to be decoded.

12.5.4   Wake-Up Code

The wake-up code is a series of Manchester bits that can have 8 or 16 Manchester bits as defined by the LFWU8 bit. The wake-up code A with 8 bits (LFWU8 is one) is stored in LFCA0 register. The wake-up code A with 16 bits (LFWU8 is zero) is stored in LFCA1 (most significant byte) and LFCA0 (least significant byte) registers. The wake-up code B with 8 bits (LFWU8 is one) is stored in LFCB0 register. The wake-up code B with 16 bits (LFWU8 is zero) is stored in LFCB1 (most significant byte) and LFCB0 (least significant byte) registers.

The wake-up code is received most significant bit first.

If the received wake-up code matches with the wake-up code A, then the LFAF bit is set. The LFAF bit is cleared when a '1' is written in the LFDFAK bit.

If the received wake-up code matches with the wake-up code B, then the LFBF bit is set. The LFBF is cleared when a '1' is written in the LFDFAK bit.

Therefore, the MCU can determine which wake-up code was received by reading the LFAF and LFBF flag bits. If neither of the wake-up codes were matched then the LFR detector and MFO are turned off and the LFR decoder is reset for t_{RST}. If a matching code was received the LFR detector and MFO will remain powered up to allow reception of any subsequent data bytes being transmitted.

12.5.5   Data Bytes

Following a successful match in wake-up code, additional data may be decoded as 8-bit bytes. The data bytes are received most significant bit first. After the last bit in each 8-bit byte is decoded, the data will be loaded into the LFDATA register, the data ready flag LFDF will be set and the LFR interrupt will be generated (if the LFIE bit is set).

The LFR interrupt will be deactivated when the LFDF bit is cleared. The LFDF is cleared when a '1' is written in the LFDFAK bit.

The MCU can then check the LFDF flag for a data byte being received and read the 8-bit data in the LFDATA register. Subsequent 8-bit bytes of data will continue to generate new interrupts and the most current data will overwrite the contents of the LFDATA register. Therefore the LFDATA register contents will be lost if the MCU fails to read the data before the LFR decodes the next byte.

The LFR detector and MFO will remain powered up as long as there is Manchester data being received. When the LFR decoder receives a Manchester data that is not valid (Manchester code violation), it treats it as the end of LF data frame. LFR decoder sets MDDO bit, generates LFR interrupt (if the LFIE bit is set), turns off LFR detector and MFO (if the MFO clock is not also being used by the ACI or PCI at that time), and LFR decoder is reset for t_{RST}.

The LFR interrupt will be cleared when the MDDO bit is cleared. The MDDO is cleared when a '1' is written to the LFDFAK bit or a '0' is written to the LFEN bit.

12.5.6   LF Reset Time

When the reset time period, t_{RST}, is initiated the LFOFF bit will be set. When the t_{RST} time expires the LFOFF bit will be cleared. The reset time period is a multiple of the sample period that is selected by the LFRST bit.

12.6   Register Descriptions

12.6.1   LF Wake-Up Code A — Byte 0 (LFCA0) Register

	7	6	5	4	3	2	1	0
R				LFA[7:0]				
W								
Reset	0	0	0	0	0	0	0	0
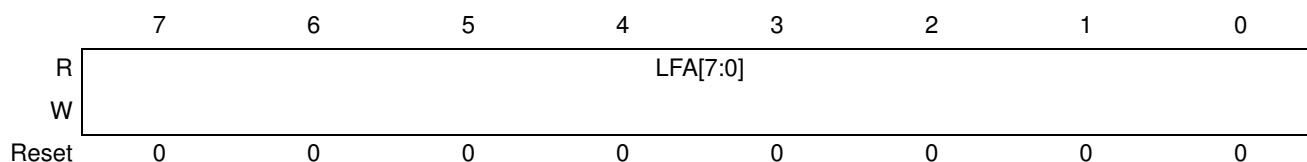
Figure 12-12 LF Wake-Up Code A — Byte 0 (LFCA0) Register

Table 12-2 LFCA0 Field Descriptions

Field	Description
7–0 LFA[7:0]	**Wake-Up Code A** — 8 least significant bits. The LFCA0 register contains the least significant byte (bits 0 until 7) of one of the wake-up codes (code A) required in a LF Manchester datagram. Write to these bits is only effective when LFEN=0.

MPXY8300 Series

12.6.2 LF Wake-Up Code A — Byte 1 (LFCA1) Register

	7	6	5	4	3	2	1	0
R				LFA[15:8]				
W								
Reset	0	0	0	0	0	0	0	0

Figure 12-13 LF Wake-Up Code A — Byte 1 (LFCA1) Register

Table 12-3 LFCA1 Field Descriptions

Field	Description
7–0 LFA[15:8]	**Wake-Up Code A** — 8 most significant bits. The LFCA1 register contains the most significant byte (bits 8 until 15) of one of the wake-up codes (code A) required in a LF Manchester datagram. Write to these bits is only effective when LFEN=0.

12.6.3 LF Wake-Up Code B — Byte 0 (LFCB0) Register

	7	6	5	4	3	2	1	0
R				LFB[7:0]				
W								
Reset	0	0	0	0	0	0	0	0

Figure 12-14 LF Wake-Up Code B — Byte 0 (LFCB0) Register

Table 12-4 LFCB0 Field Descriptions

Field	Description
7–0 LFB[7:0]	**Wake-Up Code B** — 8 least significant bits. The LFCB0 register contains the least significant byte (bits 0 until 7) of one of the wake-up codes (code B) required in a LF Manchester datagram. Write to these bits is only effective when LFEN=0.

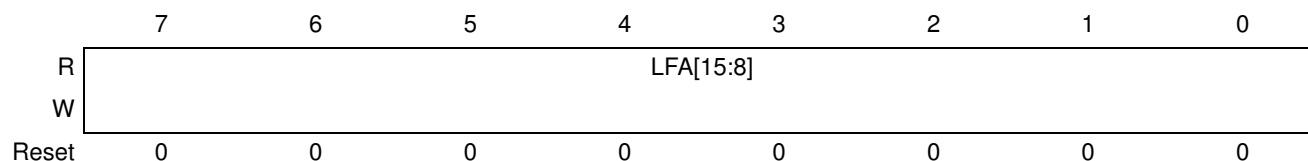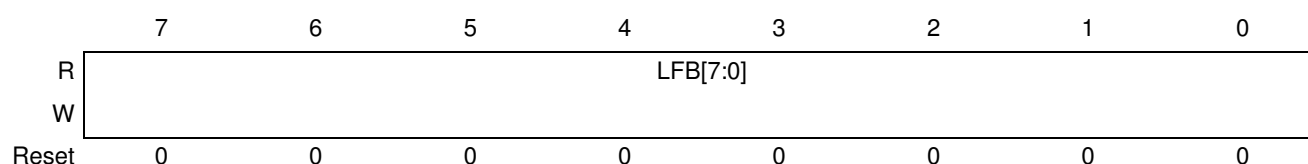### 12.6.4 LF Wake-Up Code B — Byte 1 (LFCB1) Register

	7	6	5	4	3	2	1	0
R				LFB[15:8]				
W								
Reset	0	0	0	0	0	0	0	0
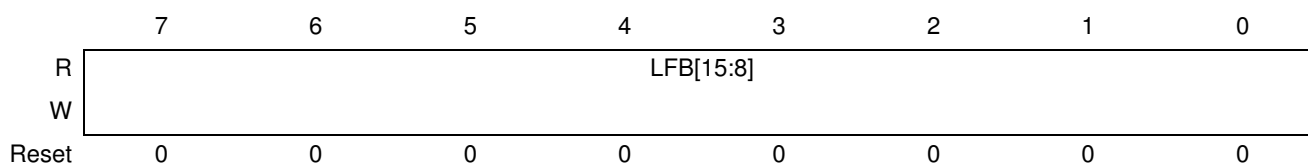
Figure 12-15 LF Wake-Up Code B — Byte 1 (LFCB1) Register

Table 12-5 LFCB1 Field Descriptions

Field	Description
7–0 LFB[15:8]	**Wake-Up Code B** — 8 most significant bits. The LFCB1 register contains the most significant byte (bits 8 until 15) of one of the wake-up codes (code B) required in a LF Manchester datagram. Write to these bits is only effective when LFEN=0.

12.6.5 LF Data Register (LFDATA)

The functions of the bits in this register are dependent on the state of the LFAGC bit in the LFCS0 register. When LFAGC is clear, a read of the LFDATA register returns:
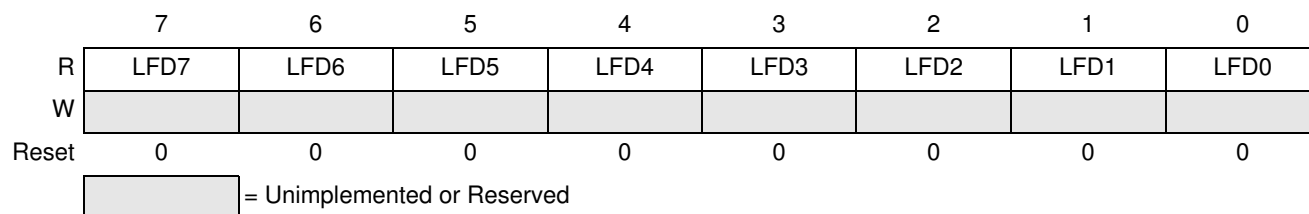
	7	6	5	4	3	2	1	0
R	LFD7	LFD6	LFD5	LFD4	LFD3	LFD2	LFD1	LFD0
W								
Reset	0	0	0	0	0	0	0	0

| = Unimplemented or Reserved |

Figure 12-16 LF Data Register (LFDATA), with LFAGC Clear
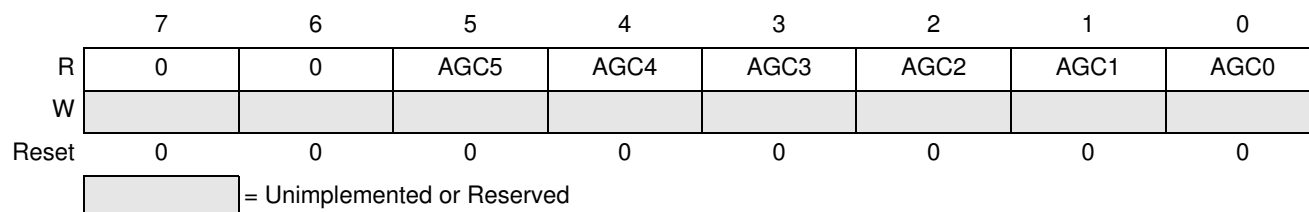
When LFAGC is set a read of the LFDATA register returns:

	7	6	5	4	3	2	1	0
R	0	0	AGC5	AGC4	AGC3	AGC2	AGC1	AGC0
W								
Reset	0	0	0	0	0	0	0	0

| = Unimplemented or Reserved |

Figure 12-17 LF Data Register (LFDATA), with LFAGC Set

Table 12-6 LFDATA Field Descriptions

Field	LFAGC	Description
7–0 LFD[7:0]	0	**LF Data Bits** — These bits are the 8-bit data contained at the end of a valid Manchester datagram in the Manchester data mode.These data bits are read when the LFAGC bit is clear. Any write to these bits is ignored.
7-6 unused	1	Unused bits, always read as 0.
5–0 AGC[5:0]		**AGC Setting** — These bits are the 6-bit setting of the AGC when using the Manchester data mode.These bits are read when the LFAGC bit is set. Any write to these bits is ignored.

12.6.6   LF Control Register (LFCR)

	7	6	5	4	3	2	1	0
R	LFM		LFCT	LFON	LFS			
W								
Reset	0	0	0	0	1	1	1	1
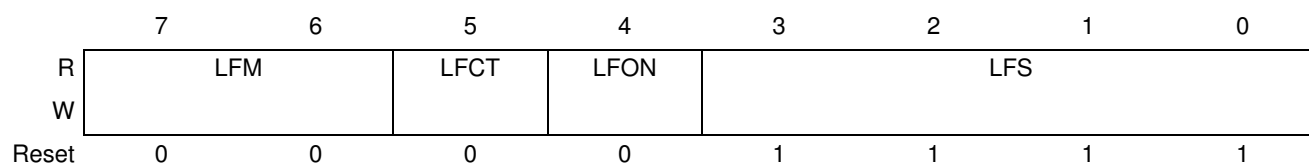
Figure 12-18 LF Control Register (LFCR)

Table 12-7 LFCR Field Descriptions

Field	Description
7–6 LFM[1:0]	**LFR Operating Mode** — The LFM0 and LFM1 bits control the decoding mode of the LFR decoder. 00 MCU Direct. 01 Carrier Detect. 1X Manchester Data.
5 LFCT	**LF Carrier Detection Time** — This bit controls the length of the carrier, t_{CD}, needed to generate the LFR interrupt in the carrier detect mode. 0 t_{CD} set at 11 periods of MFO (approximately 88µs). 1 t_{CD} set at 22 periods of MFO (approximately 176µs).

Table 12-7 LFCR Field Descriptions (Continued)

Field	Description
4 LFON	**LF Sample On Time** — This bit controls the length of time, t_{LFON}, that the LF sampling is on when in the carrier detect mode. 0 t_{LFON} set at 25 periods of MFO (approximately 200µs). 1 t_{LFON} set at 265 periods of MFO (approximately 2120µs).
3–0 LFS[3:0]	**LF Sample Reset Time** — These bits control a 4-bit reset timer driven by the LFO divided by a binary number to set the low power reset interval, t_{RST}, as given in Table 12-8. The total time between the start of each sample consists of ($t_{LFON} + t_{RST}$)

Table 12-8 LFR Reset Time — t_{RST}

LFS3	LFS2	LFS1	LFS0	LFO Divider	Nominal Reset Time (msec) (LFRST = 0)	Nominal Reset Time (msec) (LFRST = 1)
0	0	0	0	1	1	2
0	0	0	1	2	2	4
0	0	1	0	4	4	8
0	0	1	1	8	8	16
0	1	0	0	16	16	32
0	1	0	1	32	32	64
0	1	1	0	64	64	128
0	1	1	1	128	128	256
1	0	0	0	256	256	512
1	0	0	1	512	512	1024
1	0	1	0	1024	1024	2048
1	0	1	1	2048	2048	4096
1	1	0	0	4096	4096	8192
1	1	0	1	8192	8192	16384
1	1	1	0	16384	16384	32768
1	1	1	1	LFR Detector is continuously OFF		

12.6.7 LF Control/Status Register 0 (LFCS0)

	7	6	5	4	3	2	1	0
R	LFDO	LFIE	LFOFF	LFRST	LFEN	LFWU8	LFLVL	LFAGC
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved
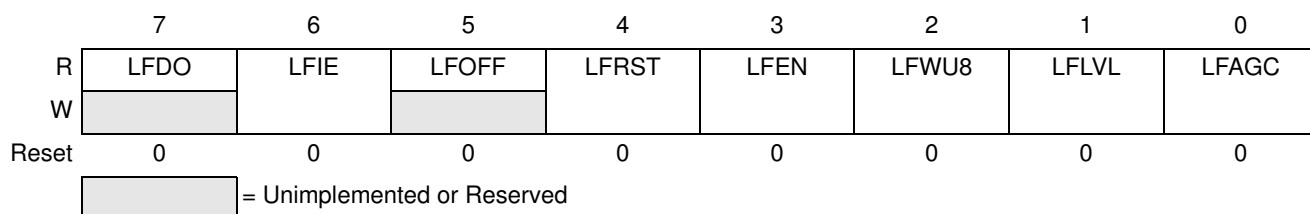
Figure 12-19 LF Control/Status Register 0 (LFCS0)

Table 12-9 LFCS0 Field Descriptions

Field	Description
7 LFDO	**LFR Detector Output** — This status bit follows the output of the LFR detector while it is sampled. If the LFR detector is not powered up the result is forced to zero. 0 Output of LFR detector output is low. 1 Output of LFR detector output is high.
6 LFIE	**LFR Interrupt Enable** — The LFIE bit enables the generation of the LFR interrupt from the LFR to MCU. 0 The generation of the LFR interrupt is disabled. 1 The generation of the LFR interrupt is enabled.
5 LFOFF	**LFR Lockout** — This status bit indicates if a Manchester encoding error has occurred and LFR detector and LFR decoder are disabled for the reset time, t_{LFRST}. 0 t_{LFRST} is no longer in progress. 1 Manchester encoding error has occurred and t_{LFRST} is still in progress.
4 LFRST	**LFR Reset Time** — The LFRST bit selects the length of time for t_{RST}. The length of time, t_{RST} is defined by counting the number of LFO rising edges equal to $(2^{LFRST})*(2^{LFS[3:0]}) +1$. The t_{RST} period ends at the start of an LFR sample time (t_{LFON}).
3 LFEN	**LFR Detector Enable** — This bit enables the LFR detector to be powered up during a given sample time. 0 LFR detector disabled. 1 LFR detector enabled during sampling.
2 LFWU8	**LFR Wake-Up Code with 8 Bits** — The LFWU8 bit selects the number of bits in Wake-Up Code (8 or 16 bits). 0 16 bit wake-up code. 1 8 bit wake-up code.
1 LFLVL	**LFR Detector Level** — The LFLVL bit selects one of two sensitivity levels for the LFR detector. 0 Detection level approximately $3.5mV_{P-P}$ (S_{DET_H}) 1 Detection level approximately $10mV_{P-P}$ (S_{DET_L})
0 LFAGC	**LFR AGC Level Select** — This bit control whether the data read from LFDATA register is a data byte of Manchester datagram or the setting of the AGC. 0 LFDATA has the setting of the AGC. 1 LFDATA has a data byte of Manchester datagram.

12.6.8 LF Control/Status Register 1 (LFCS1)

	7	6	5	4	3	2	1	0
R	LFDF	LFBPF	LFCF	0	LFBF	LFAF	MDDO	LFIG
W		LFDFAK		LFCFAK				
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-20 LF Control/Status Register 1 (LFCS1)

Table 12-10 LFCS1 Register Field Descriptions

Field	Description
7 LFDF	**LFR Data Flag** — The LFDF bit indicates when a Manchester datagram has been properly accepted and 8 bits of data have been stored to the LF data register (LFDATA). 0 LF data not ready or previously acknowledged. 1 LF data ready in the LFDATA register.
6 LFBPF	**LFR Bandpass Filter**— The read-only LFBPF bit reflects the state of the bandpass filter. 0 Filter out of range. 1 Filter in range.
6 LFDFAK	**LFR Data Flag Acknowledge** — The write-only LFDFAK bit resets the LFDF, MDDO, LFBF and LFAF flag bits. 0 No effect. 1 Reset the LFDF, LFBF and LFAF flag bits.
5 LFCF	**LFR Carrier Flag** — The LFCF bit indicates whether the received LF has lasted longer than the t_{CD} time set by the LFCT bit. 0 Carrier is no longer present and acknowledged. 1 Carrier has lasted after t_{CD}.
4 LFCFAK	**LFR Carrier Flag Acknowledge** — The LFCFAK bit resets the LFCF flag bit. 0 No effect. 1 Reset the LFCF flag bit.
3 LFBF	**Wake-Up Code Flag B** — The LFBF bit indicates when a Manchester datagram has been properly accepted and contains a wake-up code B. 0 LF wake data B not matched or previously acknowledged. 1 LF data matches wake-up code B.
2 LFAF	**Wake-Up Code Flag A** — The LFAF bit indicates when a Manchester datagram has been properly accepted and contains the wake-up code A. 0 LF wake data A not matched or previously acknowledged. 1 LF data matches wake-up code A.
1 MDDO	**Manchester Data Detect Out** — The MDDO bit gives the received Manchester data validity after the successful match in wake-up code. This bit is cleared when the LFR detector is turned off (LFEN bit is cleared). 0 It indicates that received Manchester data is valid (no Manchester code violation). 1 It indicates that received Manchester data is not valid (Manchester code violation) and therefore this is the end of the LF data frame.
0 LFIG	**Ignore LFR Detector Output** — This control bit causes the LFR decoder to ignore any sampled outputs from the LFR detector. 0 No effect. 1 The LFR detector output is held low during the LFR sample time.

12.7 Initialization Information

12.7.1 Carrier Detect Mode Initialization

In carrier detect mode the user is normally just searching for a signal of certain amplitude, carrier frequency, and duration. After this has been detected, other activity such as taking a pressure reading and/or transmitting information over an RF link can be initiated.

The wake-up code registers are not used in carrier detect mode. They can remain uninitialized. If the user needs a few extra bytes of RAM, LFCA0:1 and LFCB0:1 can be used to store variables. The LFDATA data register is not used in carrier detect mode either, but it is read only and therefore has no useless purpose in this mode.

The LFR control and status registers must be setup as defined on a bit by bit basis in Table 12-11. All control bits must be set prior to setting LFEN = 1 and enabling the module.

Table 12-11 Carrier Mode Control and Status Register Initialization

Register	Bit Position	Bit Name	Control /Status	Use and Initialization Comments	Initial Binary Value
LFCR = 0x 45	7,6	LFM1:0	C	Select carrier detect mode = 0:1.	01
	5	LFCT	C	Choice of long or short minimum character length. 0 = short (~ 88 μsec) and 1 = long (~ 176 μsec). Select short.	0
	4	LFON	C	Set length of sample time. 0 = short (~200 μsec) and 1 = long (~2120 μsec). Select short.	0
	3-0	LFS[3:0]	C	Sample rate select. Select to sample every 32 msec.	0101
LFCS0 = 0x 5A	7	LFDO	S (R only)	Does not need to be initialized.	d (0)
	6	LFIE	C	Enable interrupts. This will allow LFR to wake the CPU from Stop mode when LF module detects a valid carrier signal.	1
	5	LFOFF	S (R only)	Not applicable to carrier detect mode.	d (0)
	4	LFRST	C	Sets how long the LF module will be disabled between samples. 0 = 1 sample period, 1 = 2 sample periods.	1
	3	LFEN	C	Must be set to 1 for LFR to be enabled during sampling.	1
	2	LFWU8	C	Not applicable to carrier detect mode.	d (0)
	1	LFLVL	C	Will set for low sensitivity. If nothing is seen for a very long period, it can be switched to high sensitivity.	1
	0	LFAGC	C	Not applicable to carrier detect mode.	d (0)
LFCS1 = 0x 50	7	LFDF	S (R only)	Not applicable to carrier detect mode.	d (0)
	6	LFDFAK	C	Upon reset, LFDF, LDBF, and LFAF should be clear but it is recommended to reset them anyway.	1
	5	LFCF	S (R only)	This bit must be monitored in the carrier detect mode.	d (0)
	4	LFCFAK	C	Reset the LFCF flag.	1
	3	LFBF	S (R only)	Not applicable to carrier detect mode.	d (0)
	2	LFAF	S (R only)	Not applicable to carrier detect mode.	d (0)
	1	MDD0	S (R only)	Not applicable to carrier detect mode.	d (0)
	0	LFIG	C	Detector output must be monitored.	0

MPXY8300 Series

12.7.2 Manchester Data Mode Initialization

In Manchester data mode, several bytes of information including a synchronization character, a message ID and optional data are being received. Only the optional data packet is passed to the CPU through the LFDATA register. The data can prompt the CPU to take certain actions which might include: change sampling period, change wake-up ID, echo AGC level via RF link, take pressure reading, set critical pressure point, or transmit certain pieces of information.

The wake-up code must be initialized in Manchester mode or no information will ever be received. This must be done prior to enabling the module. The LFDATA register is read-only and does not require any initialization. The LFR control and status registers must be setup as defined on a bit by bit basis in Table 12-12. All control bits must be set prior to setting LFEN = 1 and enabling the module.

Table 12-12 Manchester Data Mode Control and Status Register Initialization

Register	Bit Position	Bit Name	Control /Status	Use and Initialization Comments	Initial Binary Value
LFCR = 0x 84	7,6	LFM1:0	C	Select Manchester data mode = 1:0	1:0
	5	LFCT	C	Not applicable to Manchester data mode.	d (0)
	4	LFON	C	No action required. In Manchester data mode, the LFON time is automatically set to 2 LFO clocks = ~2 msec.	0
	3-0	LFS[3:0]	C	Sample rate select. Choose sample every 16 msec.	0100
LFCS0 = 0x 55	7	LFDO	S (R only)	Does not need to be initialized.	d (0)
	6	LFIE	C	Enable interrupts. This will allow LFR to wake the CPU from stop mode when LF module receives a data byte in Manchester datagram.	1
	5	LFOFF	S (R only)	Manchester coding error has occurred and LF module is between sampling intervals.	d (0)
	4	LFRST	C	This sets how long the LF module will be disabled between samples. 0 = 1 sample period, 1 = 2 sample periods.	1
	3	LFEN	C	Must be set to 1 for LFR to be enabled during sampling.	1
	2	LFWU8	C	In Manchester mode this bit sets wake-up code length, 0 = 8 bits and 1 = 16 bits.	0
	1	LFLVL	C	Will start with low sensitivity. If nothing is seen for a very long period it can be switched to high sensitivity.	1
	0	LFAGC	C	Determines if LFDATA reads as AGC level or received data. 0 = data and 1 = AGC. Set for data.	0
LFCS1 = 0x 40	7	LFDF	S (R only)	Read only. Set when data is available in LFDATA reg.	d (0)
	6	LFDFAK	C	Upon reset, LFDF, LDBF and LFAF should be clear but it is recommended to reset them anyway.	1
	5	LFCF	S (R only)	Not applicable to Manchester data mode.	d (0)
	4	LFCFAK	C	Not applicable to Manchester data mode.	d (0)
	3	LFBF	S (R only)	Wakeup code B match.	d (0)
	2	LFAF	S (R only)	Wakeup code A match.	d (0)
	1	MDD0	S (R only)	Manchester data detect status: 0 = detector receiving data, 1 = detector no longer receiving data.	d (0)
	0	LFIG	C	Detector output which must be monitored.	0

12.7.3   Direct MCU Mode Initialization

In direct MCU mode, received information is analyzed by monitoring the LFDO bit. The detector can be configured for sensitivity level only. All other configuration choices are either not applicable or defaulted to a certain value. During periods of no activity, the module needs to be turned off and back on to perform the auto-zero sequence which takes approximately 32 μsec.

The wake-up code registers are not used and do not need to be initialized. The LFDATA register is read only and does not require any initialization. The LFR control and status registers must be setup as defined on a bit by bit basis in Table 12-13. All control bits must be set prior to setting LFEN = 1 and enabling the module.

Table 12-13   Direct MCU Mode Control and Status Register Initialization

Register	Bit Position	Bit Name	Control /Status	Use and Initialization Comments	Initial Binary Value
LFCR = 0x 80	7,6	LFM1:0	C	Select MCU direct mode = 0:0	0:0
	5	LFCT	C	Not applicable to direct MCU mode.	d (0)
	4	LFON	C	Not applicable to direct MCU mode.	0
	3-0	LFS[3:0]	C	Not applicable to direct MCU mode.	d (0000)
LFCS0 = 0x 55	7	LFDO	S (R only)	Does not need to be initialized.	d (0)
	6	LFIE	C	Not applicable to direct MCU mode.	0
	5	LFOFF	S (R only)	Not applicable to direct MCU mode.	d (0)
	4	LFRST	C	Not applicable to direct MCU mode.	d (0)
	3	LFEN	C	Must be set to 1 for LFR to be enabled	1
	2	LFWU8	C	Not applicable to direct MCU mode.	d (0)
	1	LFLVL	C	Will start with low sensitivity. If nothing is seen for a very long period then could switch to high sensitivity.	1
	0	LFAGC	C	Not applicable to direct MCU mode.	d (0)
LFCS1 = 0x 40	7	LFDF	S (R only)	Not applicable to direct MCU mode.	d (0)
	6	LFDFAK	C	Upon reset, LFDF, LDBF and LFAF should be clear but it is recommended to reset them anyway.	1
	5	LFCF	S (R only)	Not applicable to direct MCU mode.	d (0)
	4	LFCFAK	C	Not applicable to direct MCU mode.	d (0)
	3	LFBF	S (R only)	Not applicable to direct MCU mode.	d (0)
	2	LFAF	S (R only)	Not applicable to direct MCU mode.	d (0)
	1	MDD0	S (R only)	Not applicable to direct MCU mode.	d (0)
	0	LFIG	C	Detector output which must be monitored.	0

12.8   Application Information

The LFR module is very versatile with many sampling options available. The key for lowest power consumption and reliable communications is to have a coordinated transmit and receive interval strategy. The transmitter must broadcast a message often enough for the periodically sampling LFR to receive it in a certain time frame. The transmitter and receiver are asynchronous and controlling where in a datagram the LFR samples is not predictable. This section provides some example communication strategies along with some explanation on how software should be written to handle the events.

12.8.1   Example 1: Manchester Data Mode

An example of the sampling of a Manchester data packet is given in Figure 12-21.

In Manchester mode, a series of Manchester encoded 0s, called the preamble, starts the transmission to allow the receiver to acquire the signal and data rate. After the preamble is complete, a synchronization pulse train is transmitted to further allow the receiver to hone the data rate boundary limits and signal the start of the data frame. The data frame consists of an identification code and a variable number of data bytes.

It is critical that the LFR wakes up in the preamble information to acquire the signal. In this example, with sample period = 32 ms, the transmitter must broadcast a preamble for a minimum of 32 ms plus 8 data bit times for a message to be consistently received.

MPXY8300 Series

If the transmitted preamble is less than 32 ms plus 8 data bit times, it is possible that the LFR would wake-up in either the sync character or in the data frame. In either of these cases, the LFR will not recognize the message and will not wake-up the MCU.

The following is the equation for calculating the average current in Manchester data mode:

$$i_{lfr\_avg} = (t_{On}/t_{Period}) \times i_{lfr\_active-typ}$$

which resolves to:

$$i_{lfr\_avg} = (2\ ms\ /\ 32\ ms) \times 93\ \mu A = 5.8\ \mu A$$

DATA	>56 ms
SAMPLE TIME	32 ms
ON TIME	2000 μs
MODE	MAN DATA

Preamble is 44 ms

TIME

Tx SIGNAL — DATA 56 ms

SAMPLE

32 ms ± 30%
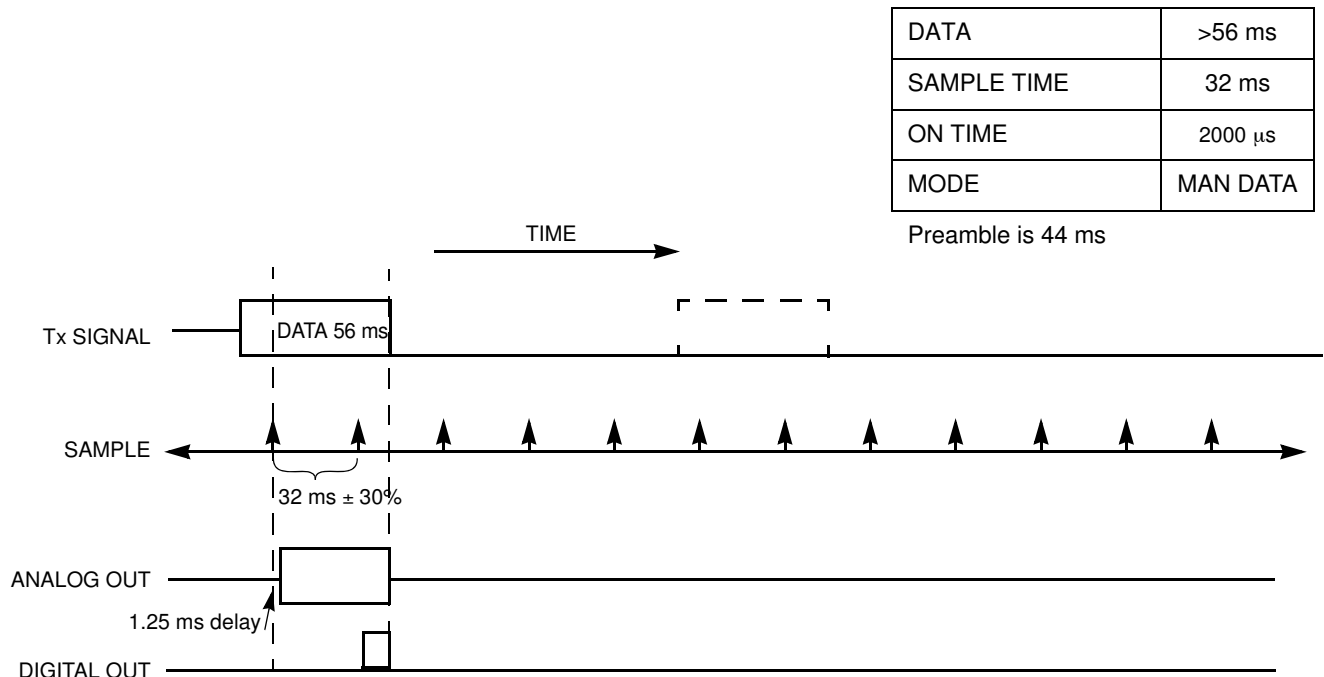
ANALOG OUT

1.25 ms delay

DIGITAL OUT

Figure 12-21 Manchester Data Mode Example Timeline

12.8.2   Example 2: Carrier Detect Mode

In this example the transmitter broadcasts either: a continuous carrier or manchester zeros for a two second interval. The LFR is initialized to sample every 1024 msec for the short LFON time of 200 μs. Referencing Figure 12-22, the first sample occurs prior to the transmitter being turned on. There is a 60-70 μs delay while the detector performs the auto-zero procedure immediately after this, the LFR samples for 128 μs. Because no signal was detected, the LFR is turned off and waits for the next sample interval. At sample #2, there is the same 60-70 μs delay while the detector auto-zeros. When the actual sample window begins, a signal is detected. The LFR monitors the signal to determine if it meets t_{CD_DET} criteria. After the criteria has been met, an interrupt is signaled.

MPXY8300 Series

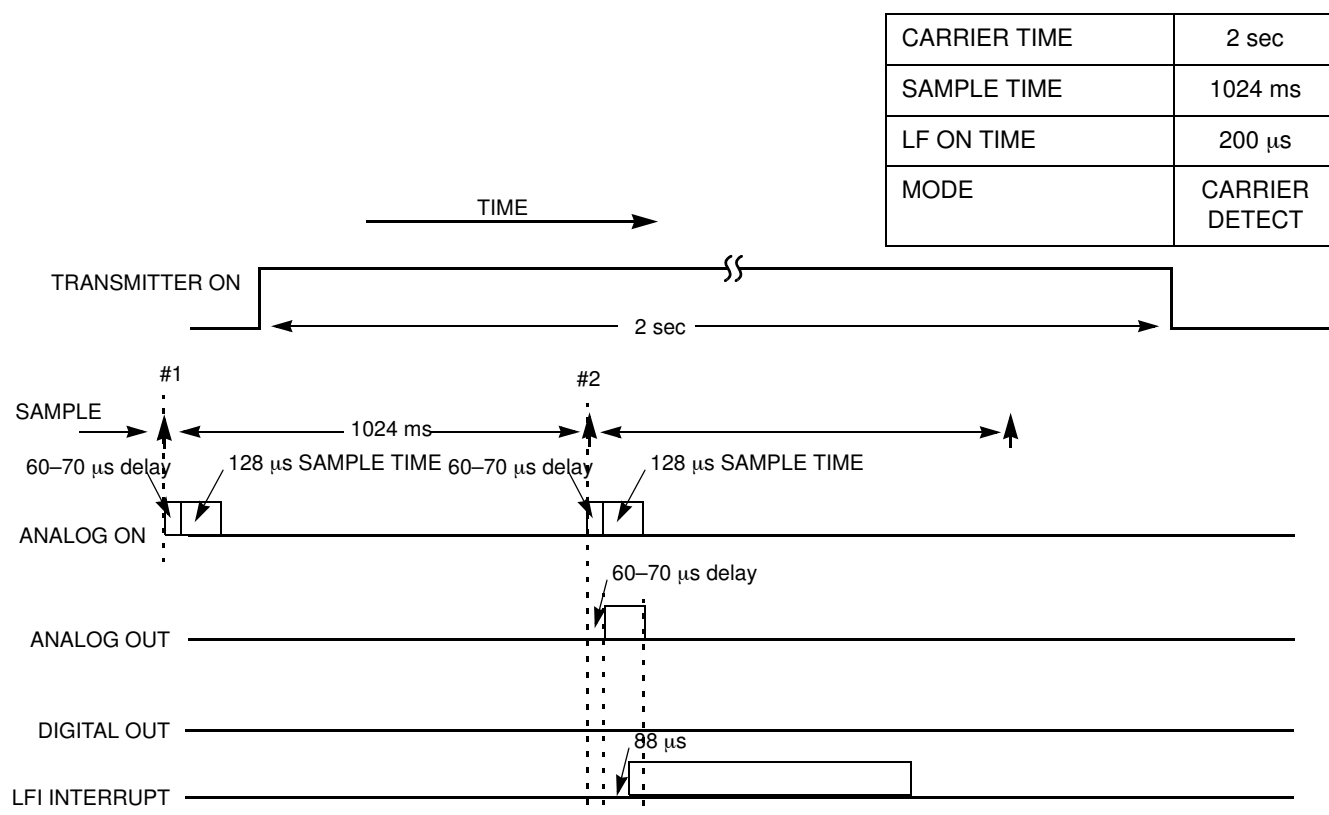| CARRIER TIME	2 sec
SAMPLE TIME	1024 ms
LF ON TIME	200 µs
MODE	CARRIER DETECT

Figure 12-22 Carrier Detect Mode Example Timeline

The interrupt service routine should disable the LFR module and clear the LFCF flag. Failure to disable the module may result in another interrupt occurring prior to the two second transmission subsiding. Other activities can be initiated inside the interrupt subroutine as dictated by the specific application.

12.9   Alternate Use of LFR Inputs

If the LFR input is not being used the associated pins default to standard MCU I/O port pins which can be configured as:

- High impedance digital inputs with optional pull-up devices
- Output digital drivers

Refer to the Section 6 for more details on how to configure and use these pins as I/O port pins.

NOTE

In all cases when using the LFR pins as general purpose inputs there will be a resistive load to V_{SS} of approximately 500 kohm. It is also recommended to keep the LFEN bit reset to '0' if using the LFR pins as general I/O.

SECTION 13   RF OUTPUT

The RFX device consists of an RF output driver for an antenna and a hardware data buffer for automated output or direct control from the MCU. It also has a charge pump to generate a higher supply voltage for the RF transmission, if desired, when the supply battery voltage is too low.

13.1   RFX Registers

The RFX device has its own memory map containing 32 register locations as shown in Figure 13-1. These registers contain control and status bits for the RFX modules, data locations for the RF data buffer, trim variables and test registers. Access to these bits is through the internal SPI interface. The firmware subroutines control this interface so that the use only needs to call the REIMS_RFRD or REIMS_RFWRT subroutines to pass the address and data that needs to be accessed or changed in the RFX. The register and bit assignments for the RFX addresses are given in Table 13-1.

NOTE

It is recommended that the firmware subroutines be used instead of trying to directly access the SPI interface.

The serial data transfers between the MCU and the RFX can be done using subroutine calls to the FLASH firmware as described in Section 14. Some subroutines will transfer information held in the CPU registers (accumulator and index register) and others will transfer blocks of data stored to specified locations in RAM, stored in the parameter registers or contained in the FLASH firmware (as calibration data).
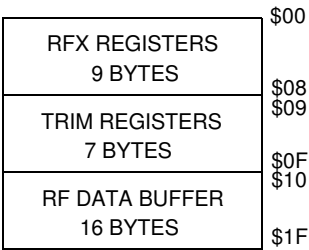
RFX REGISTERS 9 BYTES	$00
	$08
TRIM REGISTERS 7 BYTES	$09
	$0F
RF DATA BUFFER 16 BYTES	$10
	$1F

Figure 13-1 RFX Memory Map

The register and bit assignments for the RFX addresses are given in Table 13-1.

Table 13-1 RFX Register Summary

RFX Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
00	RFCR0	BPS							
01	RFCR1	EOM	FRM						
02	RFCR2	RFM	POL	CODE	CPT		AVDD	VCAP	CPUMP
03	XMTCR	SEND	RFEF	RCTS	PWR				
04	PLLCR0	AFREQ[12:5]							
05	PLLCR1	AFREQ[4:0]					1	0	0
06	PLLCR2	BFREQ[12:5]							
07	PLLCR3	BFREQ[4:0]					CF	MOD	CKREF
08	RFXTRIM1	0	0	DISCHARGE	0	0	0	0	1
09 - 0F	Unused	—	—	—	—	—	—	—	—
10	BUFF0	RFD[7:0]							
11	BUFF1	RFD[15:8]							
12	BUFF2	RFD[23;16]							
13	BUFF3	RFD[31:24]							
14	BUFF4	RFD[39:32]							
15	BUFF5	RFD[47:40]							
16	BUFF6	RFD[55:48]							
17	BUFF7	RFD[63:56]							
18	BUFF8	RFD[71:64]							
19	BUFF9	RFD[79:72]							
1A	BUFFA	RFD[87:80]							
1B	BUFFB	RFD[95:88]							
1C	BUFFC	RFD[103:96]							
1D	BUFFD	RFD[111:104]							
1E	BUFFE	RFD[119:112]							
1F	BUFFF	RFD[127:120]							

13.2 RF Data Modes

There are two modes of operation in using the RF output based on the RFM control bit as given in Table 13-2.

Table 13-2 RF Operational Modes

RFM	Mode	Data Source	Data Encoding	MCU Control	Data Rate
0	MCU Direct	Software	Software	Direct Drive of RF Output Stage	Controlled by MCU and Internal 8 MHz Oscillator
1	RF Data Buffer	Hardware	Hardware	Stores Data and Triggers Transmission	Controlled by Bit Rate Generator and External Crystal

MPXY8300 Series

13.2.1 RF Data Buffer Mode

In the RF data buffer mode the transmissions are sent by dedicated logic hardware while the MCU can be put into a low power mode until the transmission is completed.

The RF data buffer consists of a small dedicated controller and a 128-bit data buffer. The RF data buffer is loaded with whatever data pattern the user software creates. The number of data bits to be sent is selected by the FRM0:6 control bits. The control logic is triggered by the SEND control bit when it is time to transmit the data and the complete data stream is sent to the RF stage on the DATA line while being encoded as either Manchester or Bi-Phase data according to the method selected by the CODE bit.

Before the data can be transmitted the SEND control bit sets the RTS signal to enable the RF output stage to start up from its low power standby condition. The control logic will not begin the transmission phase until it receives a high input on the CTS signal from the RF stage which occurs after the RF PLL has stabilized. The type of RF transmission mode is controlled by the MOD bit as described in Section 13.5.1.

The external crystal connected to the X0 and XI pins provides the carrier frequency as well as the data rate clock, DCLK, for the data rates associated with the ASK or FSK modulation. Therefore the tolerance on the data rate will depend on the characteristics of the external crystal. One of four data rates from the bit rate generator can be selected by the BPS0:1 as described in Table 13-4.

Once the data buffer is emptied the data transfer stops; the RF output stage is turned off; and the SEND control bit is cleared. The user can test that the transmission has completed by reading back the state of the SEND control bit. If multiple frames of data are to be transmitted within a datagram the spacing between frames must be controlled by the MCU. A block diagram of the RF data buffer is shown in Figure 13-2.

NOTE

When using the data buffer mode the user's software should not change any bits in the RFX other than the SEND bit. Changing other RFX register contents can lead to data faults or errors.
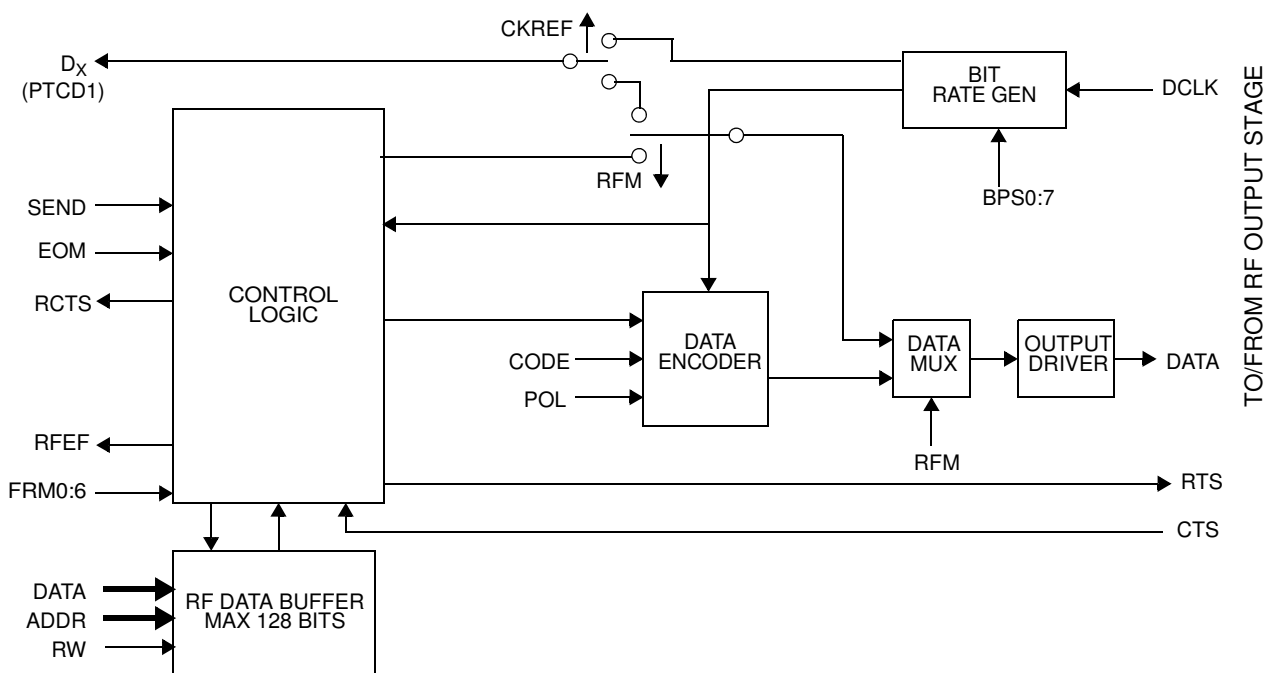

Figure 13-2 RF Data Buffer Block Diagram

13.2.2 MCU Direct Mode

In the MCU direct mode the data to the RF output stage is driven directly from the MCU. In this mode the user software must control the RF output stage to power up (using the SEND control bit), wait for the RF output stage to stabilize (monitor the RCTS status bit) and clock the DATA to the RF output stage. In this mode the data rate and its stability will depend on the internal 8 MHz oscillator.

Any transfers of data from the MCU will use the D_X signal as modulated data on the RF pin once the RF output stage is set up to transmit. The maximum data rate in this mode will depend on the complexity of the user software. The D_X signal can be controlled by the MCU PTCD1 pin internal to the MPXY8300 series package. PTCD1 can control the D_X signal to the RF output only when both the RFX CKREF and RFM bits are logical zero.

13.3   RF Output Buffer Data Frame

When using the RF data buffer mode each frame of data is sent as 1 to 128 bits per frame with a possible 2 trailing bits for an end-of-message, EOM, as shown in Figure 13-3. The actual data being transmitted in a given data frame and any combinations of data frames into a single datagram is dependent on the user software.
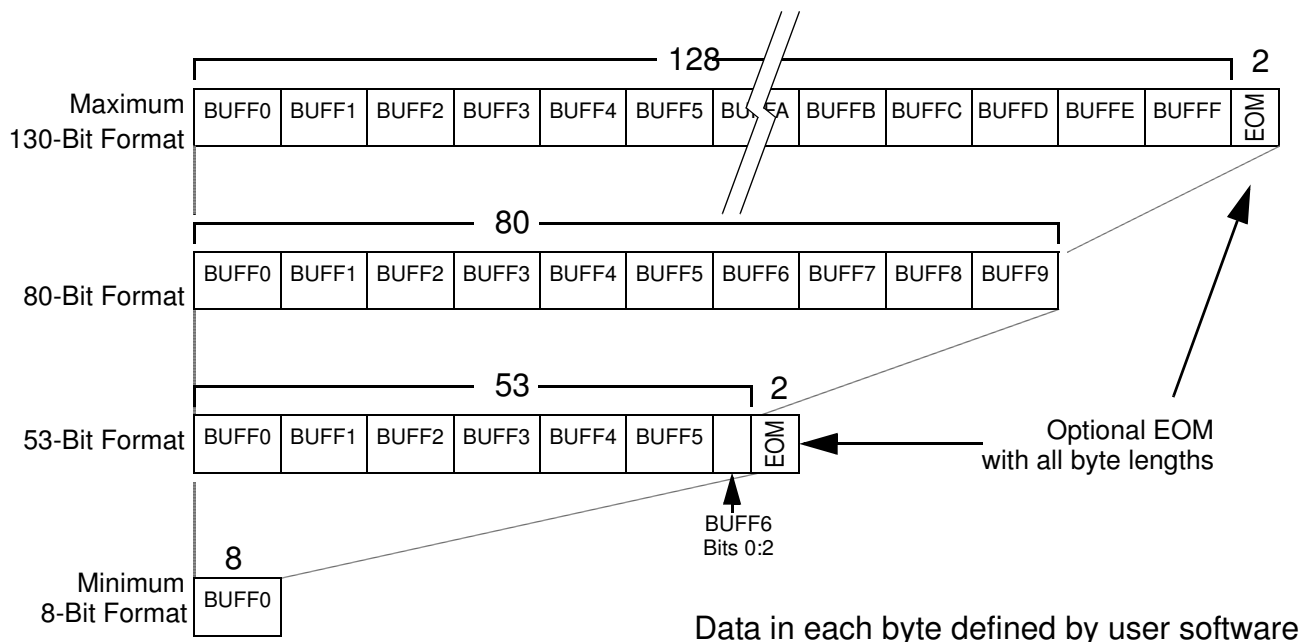

Figure 13-3 Data Frame Formats

The data buffer is unloaded to the RF output starting with the least significant bit (RFD0) in the least significant byte (BUFF0) up through the most significant bit (RFD127) in the most significant byte (BUFFF).

13.3.1   Data Buffer Length

The number of bits sent in a given transmission is selected by the FRM0:6 control bits encoded as a direct binary number plus one. This gives a range of 1 through 128 bits. Data written to data buffer bits above the highest bit number will be ignored. Transmission always begins with the data written in the BUFF0 location.

13.3.2   End of Message (EOM)

If the EOM control bit is set, then at the end of the data frame there will be carrier for a period of 2 bit times for the ASK modulation modes or $f_{RF} + \Delta f$ for the FSK modulation modes. Following the EOM period there will be no carrier for either the ASK or FSK modes. If the EOM control bit is clear, no EOM period will be added to the transmission.

13.4   Data Encoding

The CODE control bit selects either Manchester or Bi-Phase data encoding of each data bit being transferred from the RF data buffer to the RF output stage. If the CODE control bit is clear the encoding will be Manchester; and if the CODE control bit is set the encoding will be Bi-Phase. Further, the polarity of the selected encoding method can be inverted using the POL control bit.

13.4.1   Manchester Encoding

In the Manchester encoded format, data is transmitted as a transition in voltage occurring in the middle of the bit time. The polarity of this transition is selected by the POL bit. When the POL bit is cleared, then a logical LOW is defined as an increase in signal in the middle of a bit time and a logical HIGH is defined as a decrease in signal in the middle of a bit time as shown in Figure 13-4. When the POL bit is set, then a logical LOW is defined as an decrease in signal in the middle of a bit time and a logical HIGH is defined as a increase in signal in the middle of a bit time as shown in Figure 13-5. Since there is always a transition in the middle of the bit time there must also be a transition at the start of a bit time if consecutive "1" or "0" data are present.

13.4.2 Bi-Phase Encoding

In the Bi-Phase encoded format, data is transmitted as the presence or absence of a transition in signal in the middle of the bit time. The polarity of this transition is selected by the POL bit. Unlike Manchester coding there is always a signal transition at the boundaries of each bit time. When the POL bit is cleared, then a logical HIGH is defined as no change in signal in the middle of a bit time and a logical LOW is defined as a change in the signal in the middle of a bit time as shown in Figure 13-6. When the POL bit is set, then a logical HIGH is defined as a change in signal in the middle of a bit time and a logical LOW is defined as no change in the signal in the middle of a bit time as shown in Figure 13-7. Since there is always a transition at the ends of the bit time consecutive bits of the same state may have two signal states (high or low) during the middle of the bit time.

13.5 RF Output Stage

The RF output stage consists of a PLL, control logic and an output RF amplifier. Data is sent to the RF output stage from either the RF data buffer or the D_X control signal depending on the selected mode of operation as described in Section 13.2.

The RF output stage is enabled by the RTS signal which is dependent on the state of the SEND control bit. The PLL in the RF output stage will signal back via the RCTS status bit when the PLL is locked and ready to transmit. A block diagram of the RF output stage is shown in Figure 13-8. The method of modulation, carrier frequency and power output of the RF output stage can be selected as described in the following sections.
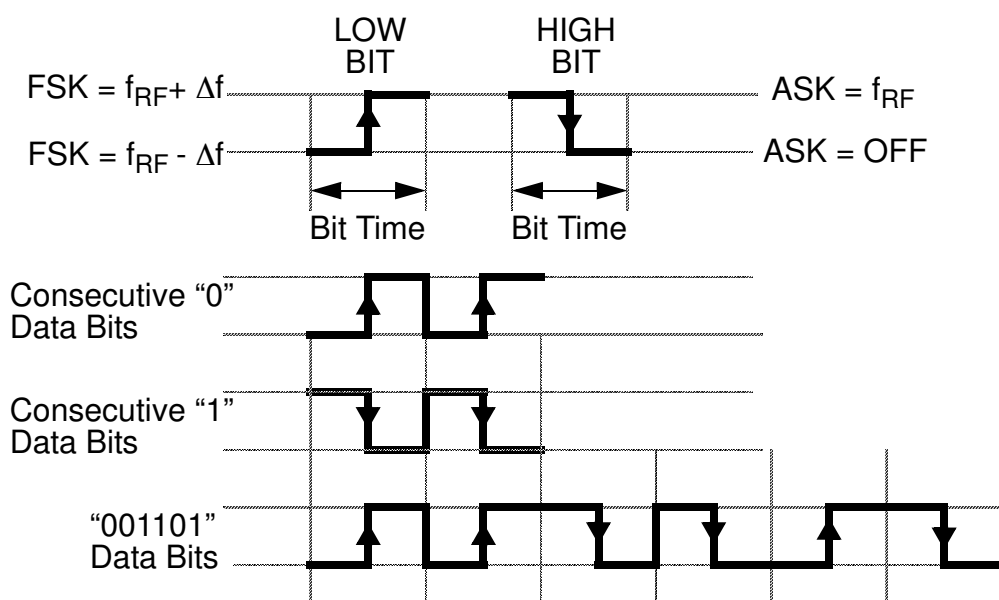

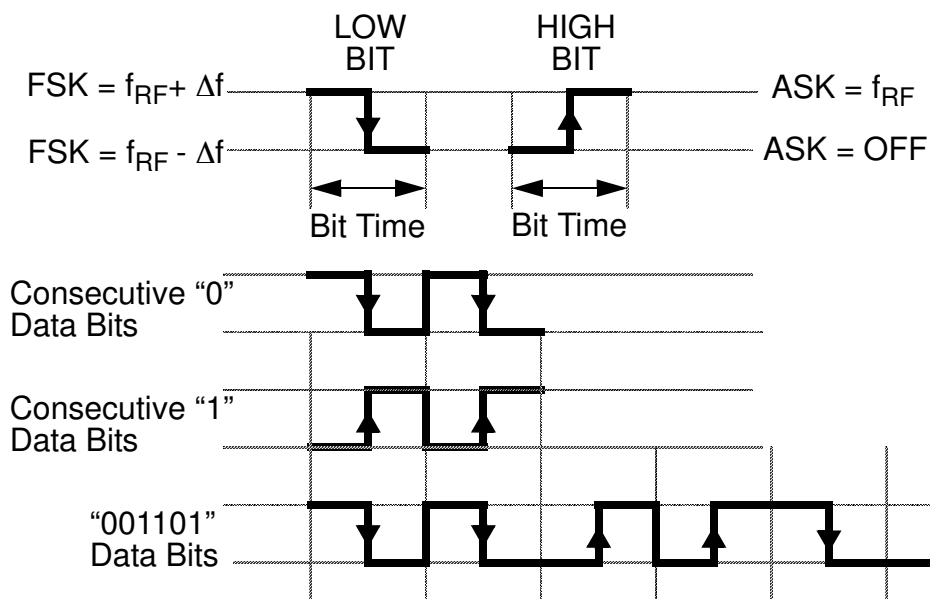**Figure 13-4 Manchester Data Bit Encoding (POL = 0)**

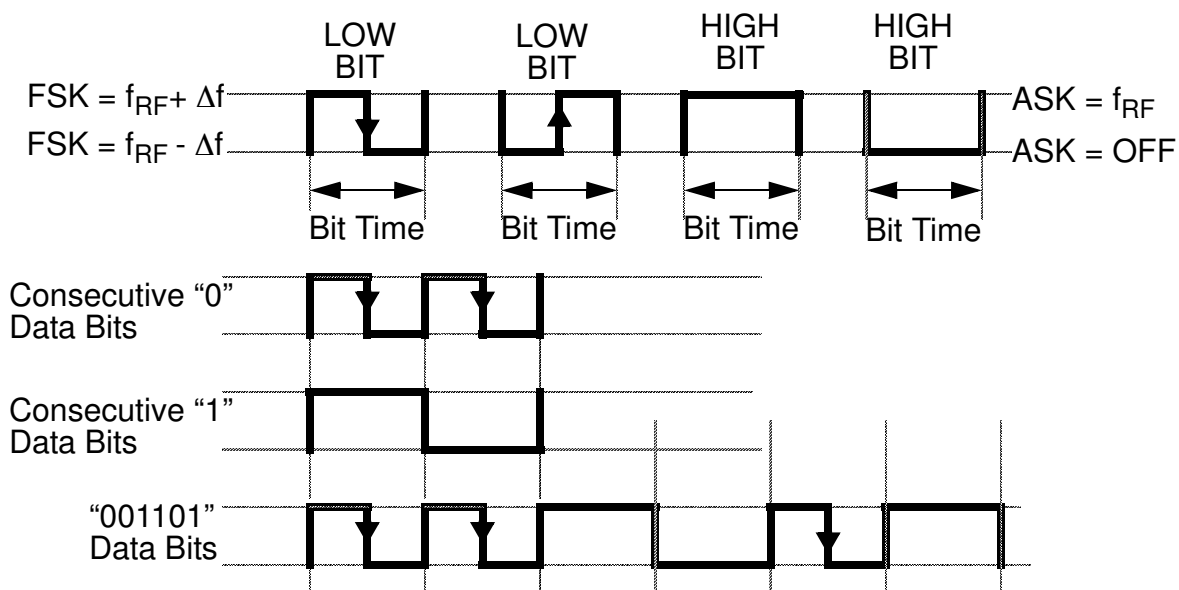**Figure 13-5 Manchester Data Bit Encoding (POL = 1)**

Figure 13-6 Bi-Phase Data Bit Encoding (POL = 0)

MPXY8300 Series
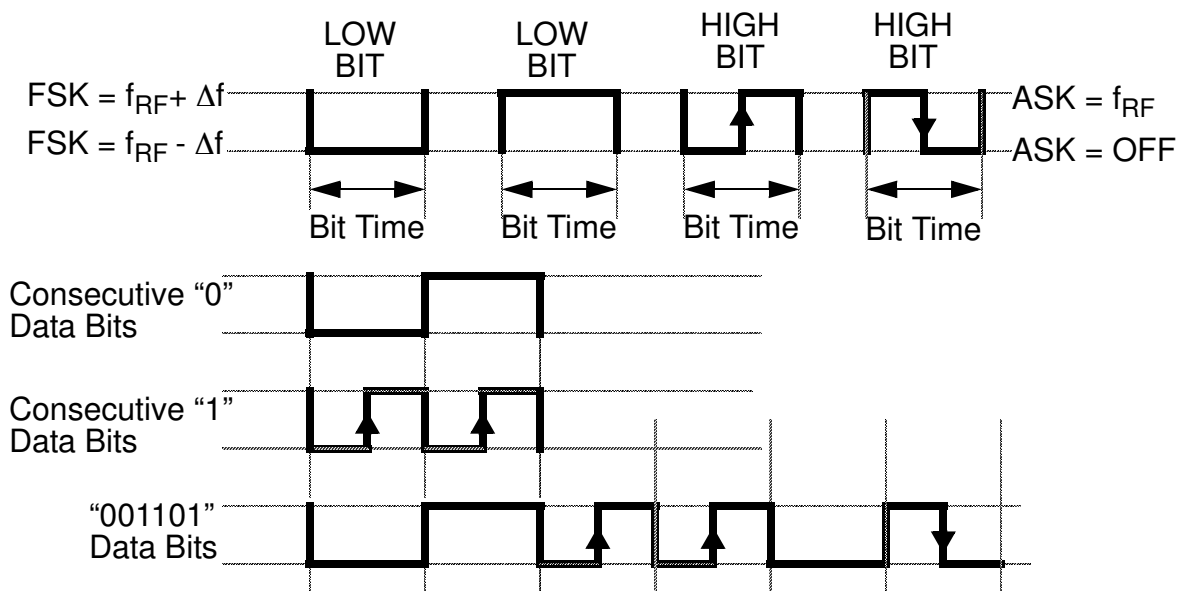
Figure 13-7 Bi-Phase Data Bit Encoding (POL = 1)
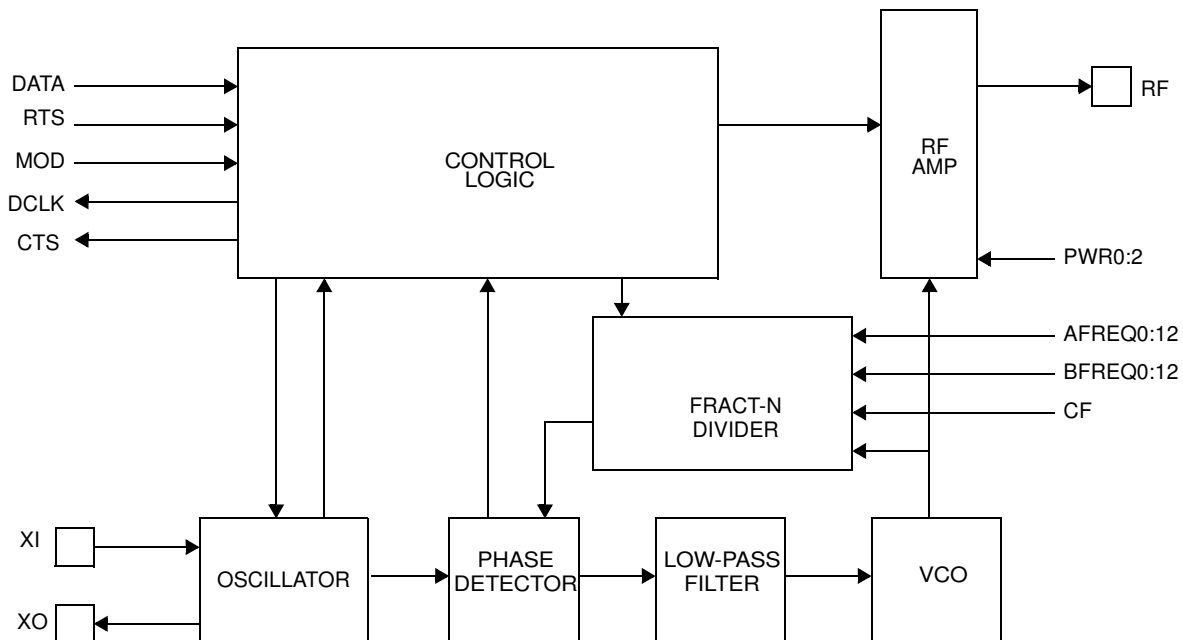
Figure 13-8 RF Output Stage Block Diagram

13.5.1   Modulation Protocol

The modulation control bit, MOD, sets the modulation of the RF signal will be either amplitude shift keying (ASK) or frequency shift keying (FSK) with several options for the frequency shift.

When operating in the FSK mode the internal, fractional-n PLL divider will be used to create the two carrier frequencies for data zero and data one. This method is more effective and robust than "pulling" the external crystal in order to shift the carrier frequency.

13.5.2 Carrier Frequency

The carrier frequency is established mainly by the external crystal used, but a centering of the fractional-n PLL provides more precise control. If the CF control bit is clear the PLL will be configured for a carrier center frequency of the 315 MHz. If the CF control bit is set the PLL will be configured for a carrier center frequency of the 434 MHz.

13.5.3 RF Power Output

The maximum power output from the RF pin can be adjusted to one of 24 levels using the PWR0:4 bits in the XMTCR.

13.5.4 Transmission Error

Any transmission will be aborted if one of the following occurs:

1. The CTS signal does not become active within the t_{LOCK} time.
2. The PLL falls out of lock after once being set and the RTS signal is still active.

If either of these cases occurs the RF output will be turned off; the SEND control bit will be cleared; and the transmission error status flag, RFEF, will be set. The RFEF bit is cleared when the SEND control bit is set again later.

13.6 Register Descriptions

13.6.1 RFX Control Register 0 - RFCR0

The RFCR0 register contains two control bits for the RFX as described in Figure 13-9. This register is accessed by writing to the RFX register address $00 using the internal SPI.

$00

	Bit 7	6	5	4	3	2	1	Bit 0
R				BPS				
W								
RESET:	0	0	0	1	1	0	0	1
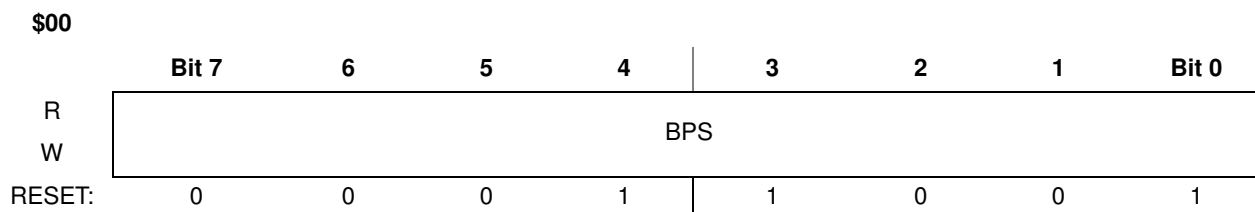
Figure 13-9 RFX Control Register 0 (RFCR0)

Table 13-3 RFX Control Register 0 Field Descriptions

Field	Description
7-0 BPS[7:0]	Data Rate — The BPS0:7 control bits select the data rate for the transmitted datagrams as described by the following equation: $$f_{DATA} = \frac{1000000 \times f_{XTAL}}{D \times (n + 1)}$$ Where: f_{DATA} = Data rate in bits/second f_{XTAL} = External crystal frequency in MHz D = Carrier selection (64 if CF bit zero, 88 if CF bit set) n = Decimal value of binary weighted BPS bits Examples of the value of n for common data rates are given in Table 13-4. The BPS bits are set to $19 by a RFX reset which results in a default data rate of approximately 9600 bps.

Table 13-4 Data Rate Options

Data Rate		BPS Value	
Target	Nominal	f_{RF} = 315 MHz and CF = 0 f_{RF} = 434 MHz and CF = 1	
		Decimal	Hexidecimal
2000 bps	2003	124	$7C
2400 bps	2408	103	$67
4000 bps	3975	62	$3E
4500 bps	4472	55	$37
4800 bps	4816	51	$33
5000 bps	5008	49	$31
9600 bps	9632	25	$19
19200 bps	19264	12	$0C

13.6.2   RFX Control Register 1 - RFCR1

The RFCR1 register contains eight control bits for the RFX as described in Figure 13-10. This register is accessed by writing to the RFX register address $01 using the SPI.

$01

|	Bit 7	6	5	4	3	2	1	Bit 0
R	EOM	FRM						
W								
RESET:	0	0	0	0	0	0	0	0
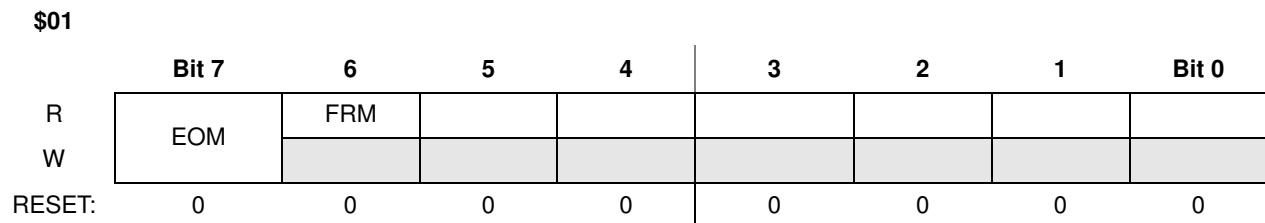
Figure 13-10 RFX Control Register 1 (RFCR1)

Table 13-5 RFX Control Register 1 Field Descriptions

Field	Description
7 EOM	End of Message — The EOM control bit selects whether there will be two data bit times of no carrier at the end of each datagram. The EOM control bit is cleared by a RFX reset. 0 EOM bit times not added 1 EOM bit times added
6-0 FRM[6:0]	Frame Bit Length — The FRM control bits select the number of bits in each datagram. The number of bits is determined by the binary value of the FRM bits plus one. This makes the range of bits from 1 to 128. The FRM control bits are cleared by a RFX reset.

13.6.3   RFX Control Register 2 - RFCR2

The RFCR2 register contains six control bits for the RFX as described in Figure 13-11. This register is accessed by writing to the RFX register address $02 using the SPI.

$02

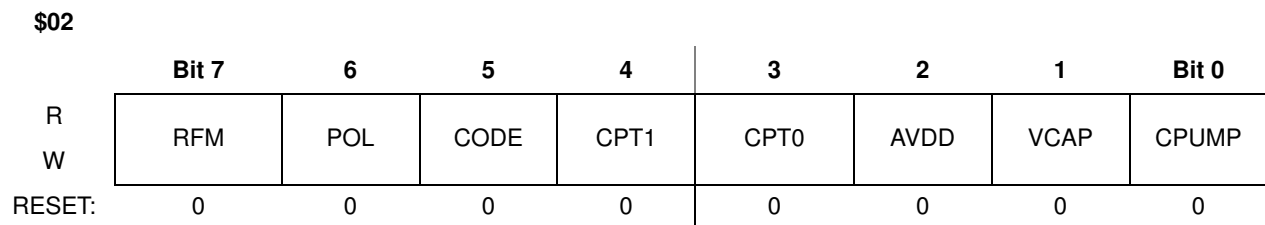|	Bit 7	6	5	4	3	2	1	Bit 0
R	RFM	POL	CODE	CPT1	CPT0	AVDD	VCAP	CPUMP
W								
RESET:	0	0	0	0	0	0	0	0

Figure 13-11 RFX Control Register 2 (RFCR2)

Table 13-6 RFX Control Register 2 Field Descriptions

Field	Description
7 RFM	**RF Modulation Source** — The RFM control bit selects the source of data to the RF power amplifier. The RFM control bit is cleared by a RFX reset. 0 Data from D_X internal pin from MCU 1 Data from RFX data buffer
6 POL	**Data Polarity** — The POL control bit selects the polarity of the data encoding for Manchester or Bi-Phase data output. The POL control bit is cleared by a RFX reset. 0 Manchester encoding polarity as in Figure 13-5 and Bi-Phase encoding polarity as in Figure 13-7 1 Manchester encoding polarity as in Figure 13-4 and Bi-Phase encoding polarity as in Figure 13-6
5 CODE	Data Encoding — The CODE control bit selects the type of data encoding. The CODE control bit is cleared by a RFX reset therefore defaulting to Manchester encoding. 0 Manchester encoding as in Figure 13-4 and Figure 13-5 1 B-Phase encoding as in Figure 13-6 and Figure 13-7
4-3 CPT[1:0]	**Charge Pump Time** — The CPT control bits select the length of time that the charge pump is on. These times are derived from the charge pump oscillator, CFO, which runs at a nominal 10 MHz frequency with a tolerance of ± 30%. The CPT control bits are cleared by a RFX reset. 00 Charge pump time is 30 msec 01 Charge pump time is 60 msec 10 Charge pump time is 120 msec 11 Charge pump time is 240 msec
2 AVDD	**RF Supply Select 2** — This control bit connects the V_{RF} pin to the AV_{DD} supply voltage. The AVDD and VACP bits should not both be set to one. This bit is set following an RFX reset. 0 V_{RF} pin disconnected from the AV_{DD} pin 1 V_{RF} pin connected to the AV_{DD} pin
1 VCAP	**RF Supply Select 1** — The VCAP control bit connects the V_{RF} pin to the source of supply voltage on the V_{CAP} pin. The AVDD and VACP bits should not both be set to one. The VCAP control bit is cleared by a RFX reset. 0 V_{RF} pin disconnected from the V_{CAP} pin 1 V_{RF} pin connected to the V_{CAP} pin
0 CPUMP	**Charge Pump Enable** — The CPUMP control bit enables the charge pump on the RFX to build up the voltage on the external capacitor connected to the V_{CAP} pin. The CPUMP bit remains set until the charge pump delay time, t_{CHG}, has elapsed or the V_{CAP} voltage limit detector has been tripped when the voltage on the V_{CAP} pin reaches V_{CAPMAX}. The CPUMP control bit can be cleared by writing a zero to it or by a RFX reset. 0 Charge pump disabled 1 Charge pump enabled and running

13.6.4 RFX Transmit Control Register - XMTCR

The XMTCR register contains six control bits and two status bits for the RFX as described in Figure 13-12. This register is accessed by writing to the RFX register address $03 using the SPI.

$03	Bit 7	6	5	4	3	2	1	Bit 0
R W	SEND	RFEF	RCTS	PWR4	PWR3	PWR2	PWR1	PWR0
RESET:	0	0	0	0	0	0	0	0

Figure 13-12 RFX Transmit Control Register (XMTCR)

Table 13-7 RFX Transmit Control Register Field Descriptions

Field	Description
7 SEND	**Transmission Start Command** — The SEND control bit starts the transmission of data held in the RFX data buffer according to the bit length specified by the FRM bits. The SEND control bit is automatically cleared when the data buffer transmission has ended or by a RFX reset. 0 Data transmission ended or transmission not in progress 1 Start data transmission or transmission in progress
6 RFEF	**RF Transmission Error Flag** — The RFEF status bit indicates if there was an error in the current of prior RF transmission as described in Section 13.5.4. The RFEF status bit is cleared by writing a logical one to the SEND bit or by a RFX reset. 0 No RF transmission error occurred 1 RF transmission error occurred
5 RCTS	**RF Ready to Send** — The RCTS status bit indicates if the VCO and PLL have started and locked and the RFX is ready to send transmit data as described in Section 13.2.2. The RCTS status bit is cleared by a RFX reset. 0 RFX not ready to send data 1 RFX ready to send data
4-0 PWR[4:0]	**RF Amplifier Power Level** — The PWR control bits select the optimum power output of the RF power amplifier. These power output levels assume optimal matching network to the RF pin. The PWR control bits are cleared by a RFX reset. The PWR control bits should normally be set to $0E. This nominal setting is targeting +5 dBm power output including the matching network loss. The LSB corresponds to a tuning of approximately 0.5 dBm and allows the user to optimize the RF link and power consumption. It can also be used to compensate the temperature and voltage variation in power output. Note that settings above nominal setting of $0E, the output power amplifier compression is limiting the effective output power.

13.6.5   PLL Control Registers A- PLLCR0:1

The PLLCR0:1 registers contain 13 control bits for the RFX as described in Figure 13-13. This register is accessed by writing to the RFX register address $04 and $05 using the SPI.

$04	Bit 7	6	5	4	3	2	1	Bit 0
R W				AFREQ[12:5]				
RESET:	0	0	0	0	0	0	0	0
$05								
R W			AFREQ[4:0]			1	0	0
RESET:	0	0	0	0	0	1	0	0

Figure 13-13 PLL Control Registers A (PLLCR0:1)

Table 13-8 PLL Control Registers A Field Descriptions

Field	Description
$04 [7-0] $05 [7-3] AFREQ [12:0]	**PLL Divider Ratio A** — The AFREQ control bits select the PLL divider ratio for a data zero in the FSK mode of modulation as described by the following equation: $$f_{CARRIER} = f_{XTAL} \times \left(19 + \frac{AFREQ}{8192}\right)$$ Where: $f_{CARRIER}$ = RF Carrier frequency in MHz f_{XTAL} = External crystal frequency in MHz AFREQ = Decimal value of the AFREQ binary weighted bits The AFREQ control bits are cleared by a RFX reset.
$05 [2] Set to 1	This bit must always be set to a 1 when writing to the PLLCR1 register.
$05 [1-0] Set to 0	These bits must always be set to a 0 when writing to the PLLCR1 register.

13.6.6   PLL Control Registers B- PLLCR2:3

The PLLCR2:3 registers contain 13 control bits for the RFX as described in Figure 13-14. This register is accessed by writing to the RFX register address $04 and $05 using the SPI.

$06	Bit 7	6	5	4	3	2	1	Bit 0
R W	BFREQ12	BFREQ11	BFREQ10	BFREQ9	BFREQ8	BFREQ7	BFREQ6	BFREQ5
RESET:	0	0	0	0	0	0	0	0
$07								
R W	BFREQ4	BFREQ3	BFREQ2	BFREQ1	BFREQ0	CF	MOD	CKREF
RESET:	0	0	0	0	0	0	0	0

Figure 13-14 PLL Control Registers B (PLLCR2:3)

Table 13-9 PLL Control Registers B Field Descriptions

Field	Description

$06 [7-0]
$07 [7-3]
BFREQ
[12:0]
	PLL Divider Ratio B — The BFREQ control bits select the PLL divider ratio for a data one in either the ASK or FSK modes of modulation as described by the following equation:
	
	$$f_{CARRIER} = f_{XTAL} \times \left(19 + \frac{BFREQ}{8192}\right)$$
	
	where:
	  $f_{CARRIER}$ = RF Carrier frequency in MHz
	  f_{XTAL} = External crystal frequency in MHz
	  BFREQ = Decimal value of the BFREQ binary weighted bits
	
	The BFREQ control bits are cleared by a RFX reset.

$07 [2] CF	**Carrier Frequency** — The CF control bit selects the optimal VCO setup and correct divider for the 500 kHz reference clock to the MCU on D_X based on the external crystals required for the desired carrier frequency. For either carrier frequency the external crystal must also be selected as shown in. The CF control bit is cleared by a RFX reset. 0 Configured for 315 MHz, PLL Divider Ratio (f_{RF}/f_{XTAL}) = 19.6713, required external crystal = 16.0132 MHz 1 Configured for 434 MHz, PLL Divider Ratio (f_{RF}/f_{XTAL}) = 19.6713, required external crystal = 22.0586 MHz
$07 [1] MOD	**RF Modulation Method** — The MOD control bit selects the method of modulating the RF. The MOD control bit is cleared by a RFX reset. 0 Configured for ASK 1 Configured for FSK
$07 [0] CKREF	**Reference Clock Enable** — The CKREF control bit controls the output of a reference clock to the MCU via the Dx signal. It is recommended that the CKREF bit be cleared before initiating an RF transmission to reduce spurious noise. The CKREF control bit is cleared by a RFX reset. 0 Reference clock signal disabled 1 Reference clock connected to the MCU

13.6.7   PLL Trim Register - RFXTRIM1

The RFXTRIM1 register contain trim and control bits for the RFX as described in Figure 13-15. This register is accessed by writing to the RFX register address $08 using the SPI.

$08	Bit 7	6	5	4	3	2	1	Bit 0
R W	0	0	DISCHARGE	0	0	0	0	1
RESET:	0	0	0	0	0	0	0	1

Figure 13-15 RFX Trim Register (RFXTRIM1)

Table 13-10 RFXTRIM Register Field Descriptions

Field	Description
$08 [7-6] Reserved	**Reserved**
$08 [5] DISCHARGE	**Charge Pump Capacitor Discharge**— This bit controls an internal discharge path from V_{CAP} to V_{SS} of approximately 17 mV/msec from 68 μF capacitor. The DISCHARGE control bit is cleared by a RFX reset. 0 Discharge path off 1 Discharge path on
$08 [4-0] Reserved	**Reserved**

13.7 Datagram Transmission Times

In order to comply with FCC requirements in the US market the periodically transmitted datagram must be less than 1 second in length and be separated by an off time that is at least 10 seconds or at least 30 times longer than the transmission time, whichever is longer. The user software must adhere to this ruling for products intended for the US market.

13.8 Charge Pump

The RFX contains a charge pump which generates a voltage up to the maximum level of V_{CAPMAX}. A block diagram of the charge pump is shown in Figure 13-16. The charge pump is enabled by the CPUMP bit in the RFCR2 register. When CPUMP is set an internal RC charge pump oscillator (CPO) is powered up to provide the switching clock for the charge pump. An internally switched, charging capacitor charges up an external capacitor connected to the V_{CAP} pin.

The CPO output frequency, f_{CPO}, is used by a time-out timer that sets a maximum time, t_{CHG}, that is set by the CPT bits.

After the t_{CHG} expires the charge pump will be turned off and the CPUMP bit cleared. However, if the voltage on the V_{CAP} pin reaches the maximum threshold of the voltage sense, V_{CHGMAX}, then the charge pump will be turned off and the CPUMP bit cleared before the t_{CHG} time has elapsed. The MCU can check the status of the charging cycle by poling the CPUMP bit. The MCU can also terminate the charge cycle early by clearing the CPUMP bit. During the charge cycle both the AVDD and VCAP control bits should be clear so there can be no reverse current back to AV_{DD} if the voltage on the V_{CAP} pin is charged above the AV_{DD} pin.

After the charge pump has completed its charging cycle the charge on the external capacitor can be used by the RF power amplifier by switching its supply voltage on the V_{RF} pin using the VCAP bit. The VCAP bit should be reset before the next charge pump cycle is started so as to not load the V_{CAP} pin. The maximum discharge time, t_{DISCHG}, is defined as the time for the voltage on the external capacitor to decay to the minimum operating voltage, V_{CAPMIN}. This discharge time is dependent on the value of the external capacitor connected to the V_{CAP} pin and the supply current used by the RF power amplifier.

The typical timing of charge pump operation when limited by the charge time, t_{CHG}, is shown in Figure 13-17. This maximum charge time will depend on the f_{CPO} clock. The typical timing of charge pump operation when limited by the charge voltage, V_{CHGMAX}, is shown in Figure 13-18. The sequencing of the charge pump and voltage select can be done using the CHGPMP firmware routine as described in Section 14.
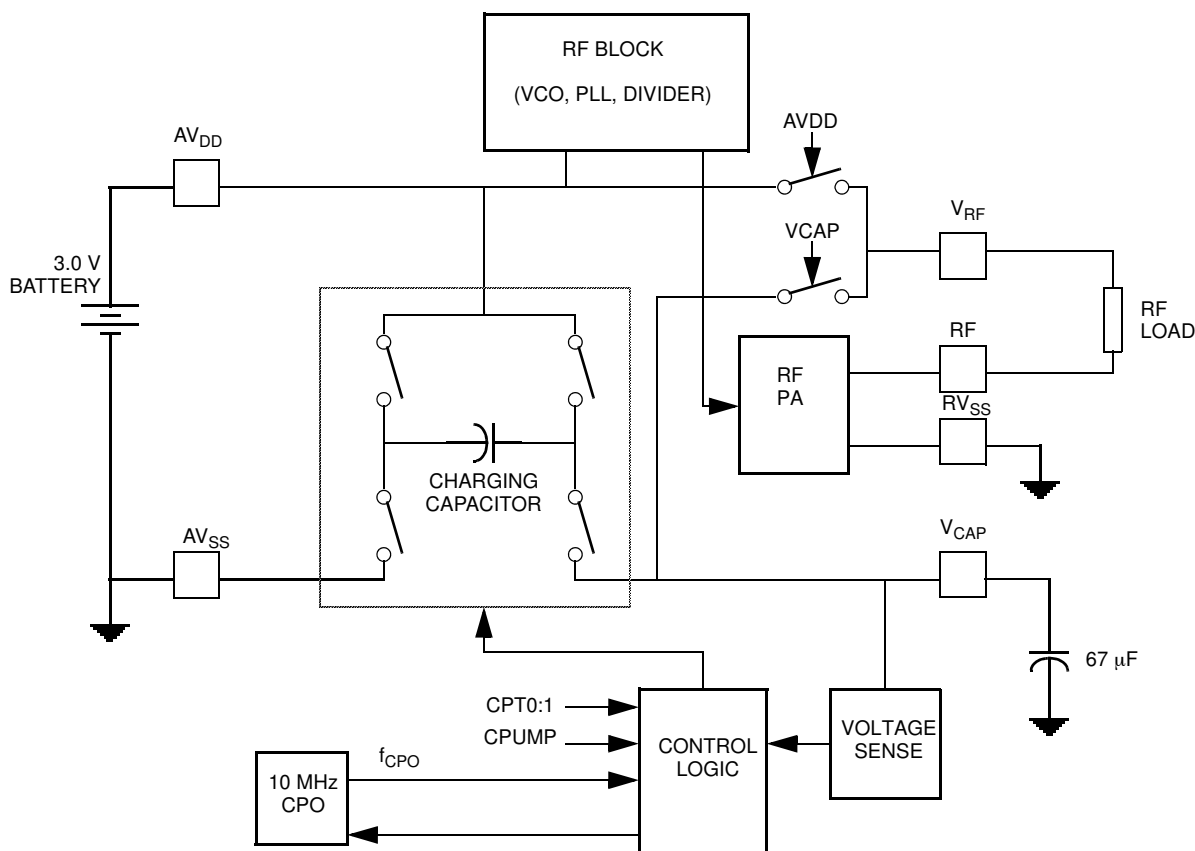
Figure 13-16 RFX Charge Pump Block Diagram
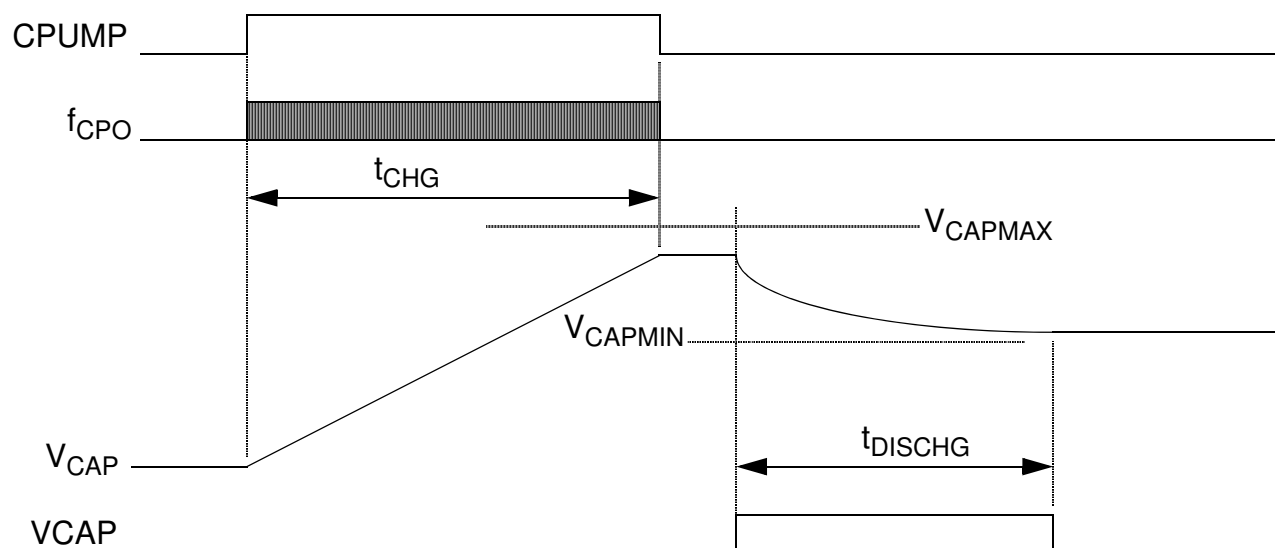
Figure 13-17 Charge Pump Timing - t_{CHG} Limit
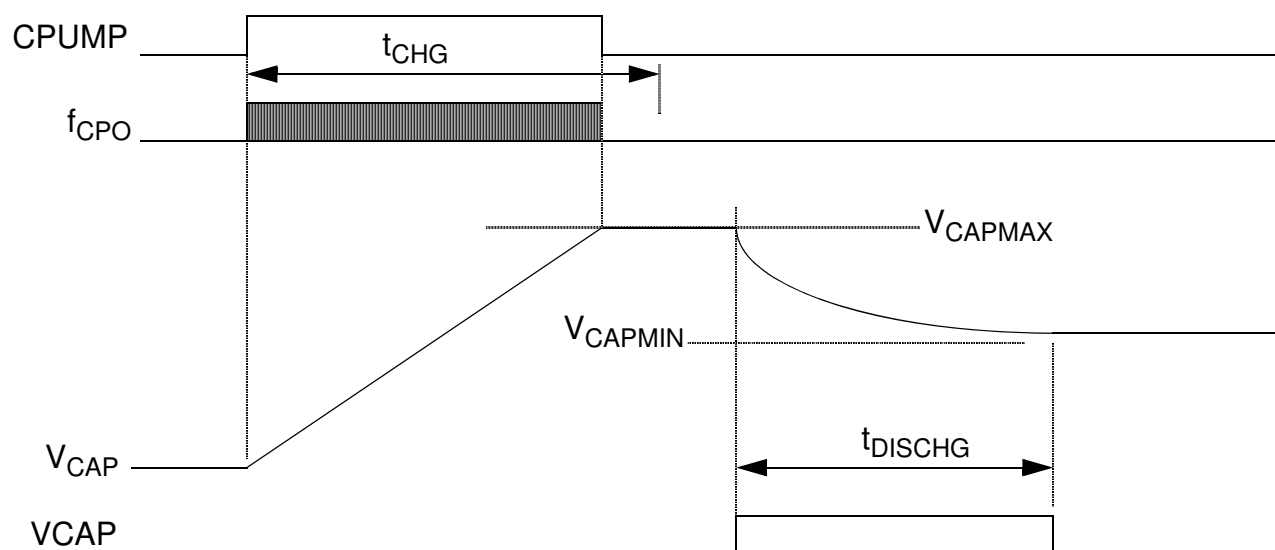
Figure 13-18 Charge Pump Timing - V_{CAP} Limit

Recommended charge pump sequence:

1.   Clear AVDD control bit to disconnect V_{RF}.
2.   Set CPUMP control bit.
3.   Wait for the CPUMP bit to be cleared by the V_{CAP} voltage reaching V_{CHGMAX} or the charge time, t_{CHG}, expires.
4.   Set VCAP control bit to connect V_{RF}.
5.   Turn on RF transmitter.
6.   Clear VCAP control bit to disconnect V_{RF}.
7.   Set AVDD control bit to connect V_{RF}.

Recommended discharge sequence for external capacitor on V_{CAP}:

1. Set VCAP control bit to connect V_{RF}.
2. Set DISCHARGE bit.
3. Wait for the desired discharge time.
4. Reset RFX.

CAUTION

If the charge pump is activated to create a voltage on the external capacitor on V_{CAP}, this charge should be used by the RF output before connecting the V_{RF} pin to the AV_{DD} source (using the AV_{DD} control bit). Otherwise the charge built up on the external capacitor may cause an excessive reverse current into the battery. Further, both the AV_{DD} and V_{CAP} control bits should not be set at the same time. Before changing between them, both control bits should first be cleared.

NOTE

If the charge pump is not going to be used the V_{CAP} pin should be connected to the AV_{DD} pin and both the VCAP and DISCHARGE control bits should remain cleared.

SECTION 14 FIRMWARE

This section describes the software subroutines contained in the firmware section of the FLASH memory that the user can call for various tasks and to reduce the software development time for the main internal operations.

14.1 Software Jump Table

All subroutines are accessed through a jump table located at the top of the FLASH firmware memory. This allows upgrades in firmware without changing the software code of the user. All subroutines should be accessed with the JSR instruction.

14.2 Function Documentation

The following subsections describe the details of the firmware routines.

14.2.1 General Rules

1. ADC is totally managed by the firmware
2. The status byte is updated by each routine. If not read between routine calls the intermediate status may be lost.
3. No output parameter can use the extreme codes (all zero's or all one's).
4. The all zero's output code will always indicate a fault and the status byte will indicate the source of the error.
5. While firmware is processing, CPU resources are unavailable for application.
6. Each measured parameter will return a limit code ($00, $FF or $1FF) if an error occurs in its acquisition.

14.2.2 Firmware Routines

The details on the use and execution of each firmware routine is documented in the Code Warrior project file that is supplied by Freescale. Any future updates to these firmware routines will be contained in that file. A summary of the firmware routines available is given in Table 14-1.

The firmware table is comprised of three-byte entries where the first byte is the operational code for the JMP instruction, and the following two bytes are the absolute address pointing to the location of the firmware function.

Table 14-1 Firmware Summary and Jump Table

Address	Routine	Description
E000	REIMS_RESET	Master reset
E003	REIMS_ACITD	Acceleration data trim
E006	REIMS_READ_COMP_VOLTAGE	Voltage acquisition
E009	REIMS_READ_COMP_TEMP_8	Temperature acquisition
E00C	REIMS_READ_COMP_PRESSURE	Pressure acquisition
E00F	REIMS_READ_COMP_ACCEL_X	X Acceleration acquisition
E012	REIMS_READ_COMP_ACCEL_Z	Z Acceleration acquisition
E015	REIMS_RF_RESET	Reset the RFX
E018	REIMS_RFRD	Read one RF register
E01B	REIMS_RF_READ_REGISTERS	Read consecutive RF registers
E01E	REIMS_RFWRT	Write one RF register
E021	REIMS_RF_WRITE_REGISTERS	Write consecutive RF registers
E024	REIMS_RF_READ_DATA	Read RFX data buffer
E027	REIMS_RF_READ_DATA_REVERSE	Read RFX data buffer, bit order transposed
E02A	REIMS_RF_WRITE_DATA	Write RFX data buffer
E02D	REIMS_RF_WRITE_DATA_REVERSE	Write RFX data buffer, bit order transposed
E030	REIMS_RF_XCO_ON	Switch on XCO
E033	REIMS_RF_SET_TX	Send RF data buffer
E036	REIMS_WAVG_2	Weighted average with R=2
E039	REIMS_WAVG_4	Weighted average with R=4

Table 14-1 Firmware Summary and Jump Table

Address	Routine	Description
E03C	REIMS_WAVG_8	Weighted average with R=8
E03F	REIMS_WAVG_16	Weighted average with R=16
E042	REIMS_WAVG_32	Weighted average with R=32
E045	REIMS_LFOCAL	LFO calibration
E048	REIMS_FLASH_WRITE	Flash write
E04B	REIMS_WIRE_CHECK	Check bounding wires
E04E	REIMS_READ_ID	Read Identification codes
E051	REIMS_LF_ENABLE	Enable LF for Carrier or Data
E054	REIMS_LF_READ_DATA	Reading LF data
E057	REIMS_SWORD_MULT	Signed multiplication
E05A	REIMS_SQUARE_ROOT	Square root
E05D	REIMS_CRC8	CRC with 8 bits polynomial
E060	REIMS_CHECKSUM_XOR	Checksum of bytes using XOR
E063	REIMS_MSG_INIT	Initialization of the serial communication
E066	REIMS_MSG_WRITE	Writing data on emulated serial interface
E069	REIMS_MSG_READ	Reading data from emulated serial interface
E06C	REIMS_READ_COMP_ACCEL_XZ	Takes X acceleration followed by Z acceleration

14.2.3   Device Identification

The bytes assigned to identify the device and its options are described below. These data can be read by use of the REIMS_READ_ID routine.

Table 14-2 Device ID Coding Summary

IDAddress	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
E0A0	CODE0	Firmware Revision							
E0A1	CODE1	REV2	REV1	REV0	PRG2	PRG1	PRG0	SUPP1	SUPP0
E0A2	CODE2	Unique Device ID							
E0A3	CODE3								
E0A4	CODE4								
E0A5	CODE5								
E0A6	CODE6								
E0A7	CODE7								
E0A8	CODE8								

Table 14-3 Device ID Coding Descriptions

Field	Description
CODE0, 7:5 REV0:2	Revision number for the multiple-chip-module from 0 to 7.
CODE0, 7:5 PRG0:2	Calibrated range for pressure and acceleration. First code of $0 for 100-800 kPa, ±10 X-axis accelerometer and 0-60g Z-axis accelerometer. Other codes to be assigned as produced.
CODE0, 1:0 SUPP0:1	Number of silicon device supplier from 0 to 3.
CODE1:4, 7:0 ID31:0	32-bit serial number for each device. All numbers to be unique, but numbering sequence to be determined by Freescale manufacturing process.
CODE5, 7:0 Reserved	**Reserved for Freescale firmware description**.

14.3   Definition of Signal Ranges

Each measured parameter (pressure, voltage, temperature, acceleration) results from an ADC conversion of an analog signal. This ADC result may then be passed by the firmware to the application software as either the raw ADC result or further compensated and scaled for an output between one and the maximum digital value minus one. The minimum digital value of zero and the maximum digital value are reserved as error codes.

The signal ranges and their significant data points are shown in Figure 14-1. In this definition the signal source would normally output a signal between S_{INLO} and S_{INHI}. Due to process, temperature and voltage variations this signal may increase its range to S_{INMIN} to S_{INMAX}. In all cases the signal will be between the supply rails, so that the ADC will convert it to a range of digital numbers between 0 and 1023. These digital numbers will have corresponding D_{INMIN}, D_{INLO}, D_{INHI}, D_{INMAX} values. The ADC digital value is taken by the firmware and compensated and scaled to give the required output code range.

Digital input values below D_{INMIN} and above D_{INMAX} are immediately flagged as being out of range and generate error bits and the output is forced to the 0 value.

Digital values below D_{INLO} (but above D_{INMIN}) or above D_{INHI} (but not D_{INMAX}) will most likely cause an output that would be less than 1 or greater than 510, respectively. These cases are considered underflow or overflow, respectively. Underflow results will be forced to a value of 1. Overflow results will be forced to a value of 510.

Digital values between D_{INLO} and D_{INHI} will normally produce an output between 1 to 510 (for a 9-bit result). In some isolated cases due to compensation calculations and rounding the result may be less than 1 or greater than 510, in which case the underflow and overflow rule mentioned above is used.
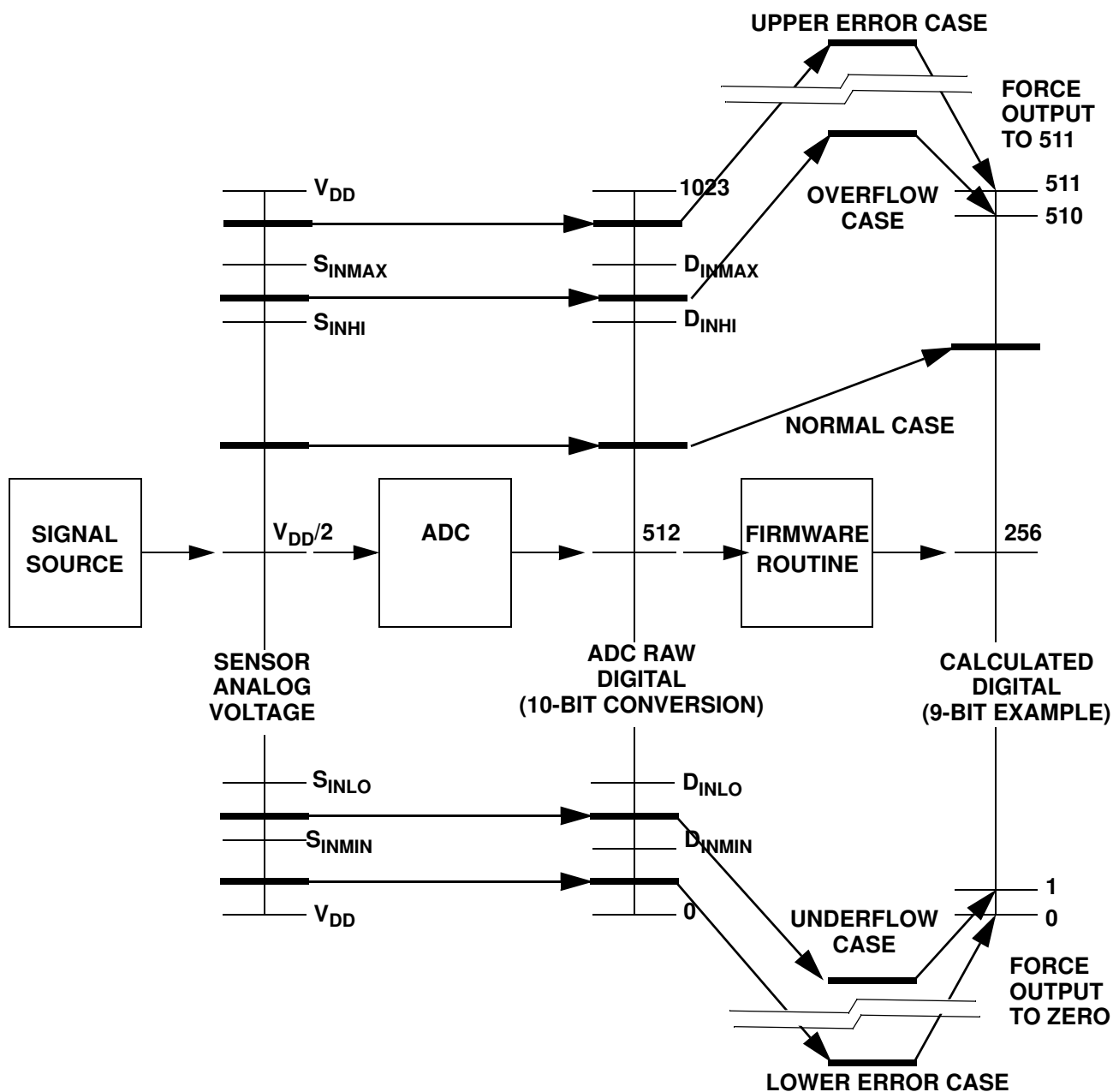
Figure 14-1 Measurement Signal Range Definitions

14.4   Firmware Memory Resource Usage

The firmware uses the top 8196 bytes of the FLASH memory map.

The firmware does not use any bytes of the RAM for global variables, but stacking intermediate values. Therefore the user should be aware of the stack depth required for each firmware routine.

The firmware uses one byte of the Parameter Registers at location $5F for interrupt flags.

SECTION 15   DEVELOPMENT SUPPORT

15.1   Introduction

This chapter describes the single-wire background debug mode (BDM), which uses the on-chip background debug controller (BDC) module, and the independent on-chip real-time in-circuit emulation (ICE) system, which uses the on-chip debug (DBG) module.

15.1.1   Features

Features of the BDC module include:
- Single pin for mode selection and background communications
- BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands for memory access
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from stop or wait modes
- One hardware address breakpoint built into BDC
- Oscillator runs in stop mode, if BDC enabled
- COP watchdog disabled while in active background mode

Features of the ICE system include:
- Two trigger comparators: Two address + read/write (R/W) or one full address + data + R/W
- Flexible 8-word by 16-bit FIFO (first-in, first-out) buffer for capture information:
  - Change-of-flow addresses or
  - Event-only data
- Two types of breakpoints:
  - Tag breakpoints for instruction opcodes
  - Force breakpoints for any address access
- Nine trigger modes:
  - Basic: A-only, A OR B
  - Sequence: A then B
  - Full: A AND B data, A AND NOT B data
  - Event (store data): Event-only B, A then event-only B
  - Range: Inside range (A ≤ address ≤ B), outside range (address < A or address > B)

15.2   Background Debug Controller (BDC)

All MCUs in the HCS08 Family contain a single-wire background debug interface that supports in-circuit programming of on-chip nonvolatile memory and sophisticated non-intrusive debug capabilities. Unlike debug interfaces on earlier 8-bit MCUs, this system does not interfere with normal application resources. It does not use any user memory or locations in the memory map and does not share any on-chip peripherals.

BDC commands are divided into two groups:
- Active background mode commands require that the target MCU is in active background mode (the user program is not running). Active background mode commands allow the CPU registers to be read or written, and allow the user to trace one user instruction at a time, or GO to the user program from active background mode.
- Non-intrusive commands can be executed at any time even while the user's program is running. Non-intrusive commands allow a user to read or write MCU memory locations or access status and control registers within the background debug controller.

Typically, a relatively simple interface pod is used to translate commands from a host computer into commands for the custom serial interface to the single-wire background debug system. Depending on the development tool vendor, this interface pod may use a standard RS-232 serial port, a parallel printer port, or some other type of communications such as a universal serial bus (USB) to communicate between the host PC and the pod. The pod typically connects to the target system with ground, the BKGD pin, $\overline{RESET}$, and sometimes V_{DD}. An open-drain connection to reset allows the host to force a target system reset, which is useful to regain control of a lost target system or to control startup of a target system before the on-chip nonvolatile memory has been programmed. Sometimes V_{DD} can be used to allow the pod to use power from the target system to avoid the need for a separate power supply. However, if the pod is powered separately, it can be connected to a running target system without forcing a target system reset or otherwise disturbing the running application program.
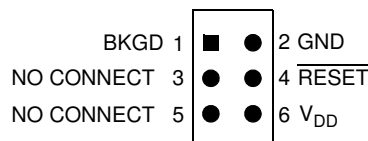

Figure 15-1 BDM Tool Connector

15.2.1   BKGD Pin Description

BKGD is the single-wire background debug interface pin. The primary function of this pin is for bidirectional serial communication of active background mode commands and data. During reset, this pin is used to select between starting in active background mode or starting the user's application program. This pin is also used to request a timed sync response pulse to allow a host development tool to determine the correct clock frequency for background debug serial communications.

BDC serial communications use a custom serial protocol first introduced on the M68HC12 Family of microcontrollers. This protocol assumes the host knows the communication clock rate that is determined by the target BDC clock rate. All communication is initiated and controlled by the host that drives a high-to-low edge to signal the beginning of each bit time. Commands and data are sent most significant bit first (MSB first). For a detailed description of the communications protocol, refer to Section 15.2.2.

If a host is attempting to communicate with a target MCU that has an unknown BDC clock rate, a SYNC command may be sent to the target MCU to request a timed sync response signal from which the host can determine the correct communication speed.

BKGD is a pseudo-open-drain pin and there is an on-chip pull-up so no external pull-up resistor is required. Unlike typical open-drain pins, the external RC time constant on this pin, which is influenced by external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speedup pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 15.2.2, for more detail.

When no debugger pod is connected to the 6-pin BDM interface connector, the internal pull-up on BKGD chooses normal operating mode. When a debug pod is connected to BKGD it is possible to force the MCU into active background mode after reset. The specific conditions for forcing active background depend upon the HCS08 derivative (refer to the introduction to this Development Support section). It is not necessary to reset the target MCU to communicate with it through the background debug interface.

15.2.2   Communication Details

The BDC serial interface requires the external controller to generate a falling edge on the BKGD pin to indicate the start of each bit time. The external controller provides this falling edge whether data is transmitted or received.

BKGD is a pseudo-open-drain pin that can be driven either by an external controller or by the MCU. Data is transferred MSB first at 16 BDC clock cycles per bit (nominal speed). The interface times out if 512 BDC clock cycles occur between falling edges from the host. Any BDC command that was in progress when this time-out occurs is aborted without affecting the memory or operating mode of the target MCU system.

The custom serial protocol requires the debug pod to know the target BDC communication clock speed.

The clock switch (CLKSW) control bit in the BDC status and control register allows the user to select the BDC clock source. The BDC clock source can either be the bus or the alternate BDC clock source.

The BKGD pin can receive a high or low level or transmit a high or low level. The following diagrams show timing for each of these cases. Interface timing is synchronous to clocks in the target BDC, but asynchronous to the external host. The internal BDC clock signal is shown for reference in counting cycles.

Figure 15-2 shows an external host transmitting a logic 1 or 0 to the BKGD pin of a target HCS08 MCU. The host is asynchronous to the target so there is a 0-to-1 cycle delay from the host-generated falling edge to where the target perceives the beginning of the bit time. Ten target BDC clock cycles later, the target senses the bit level on the BKGD pin. Typically, the host actively drives the pseudo-open-drain BKGD pin during host-to-target transmissions to speed up rising edges. Because the target does not drive the BKGD pin during the host-to-target transmission period, there is no need to treat the line as an open-drain signal during this period.

Figure 15-2 BDC Host-to-Target Serial Bit Timing

Figure 15-3 shows the host receiving a logic 1 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target MCU. The host holds the BKGD pin low long enough for the target to recognize it (at least two target BDC cycles). The host must release the low drive before the target MCU drives a brief active-high speedup pulse seven cycles after the perceived start of the bit time. The host should sample the bit level about 10 cycles after it started the bit time.
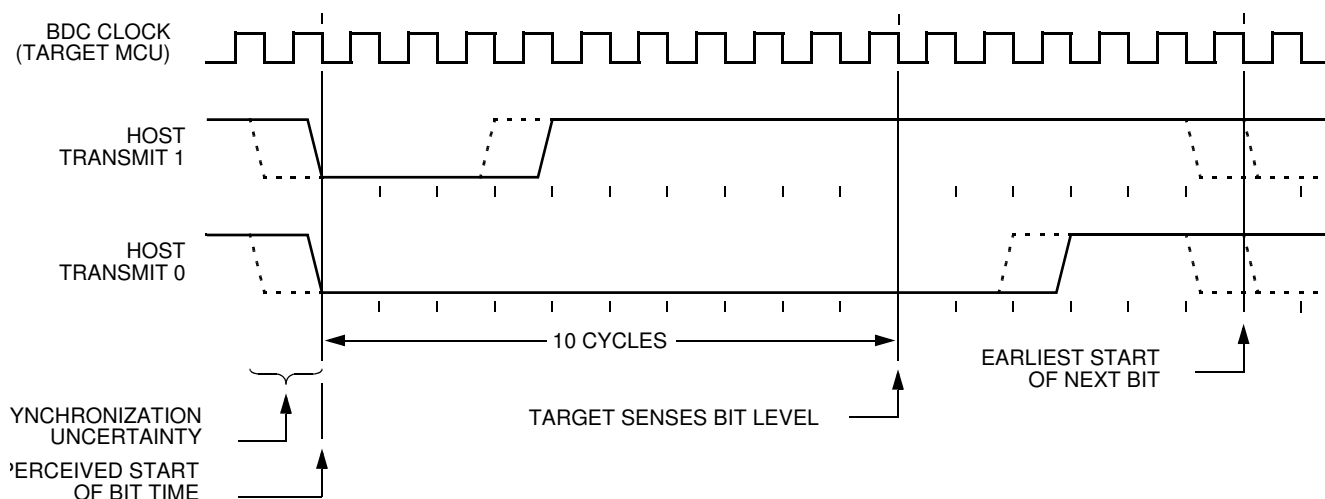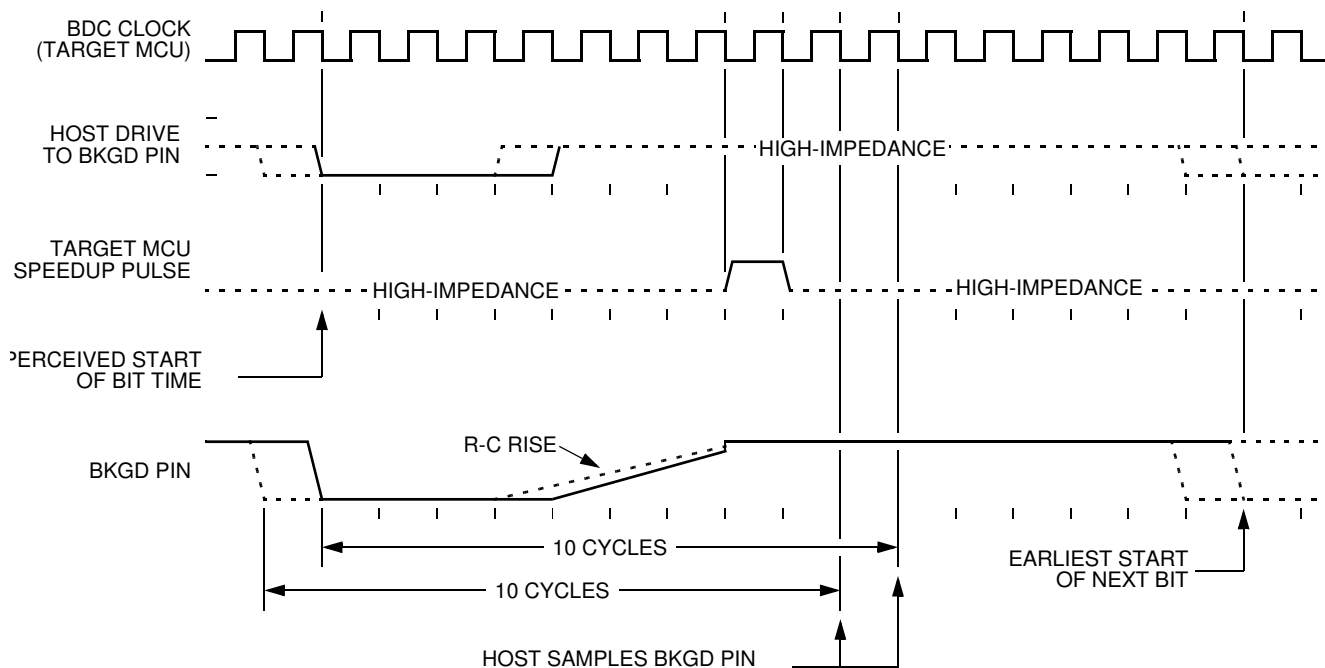

Figure 15-3 BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 15-4 shows the host receiving a logic 0 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target MCU. The host initiates the bit time but the target HCS08 finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 BDC clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 cycles after starting the bit time.
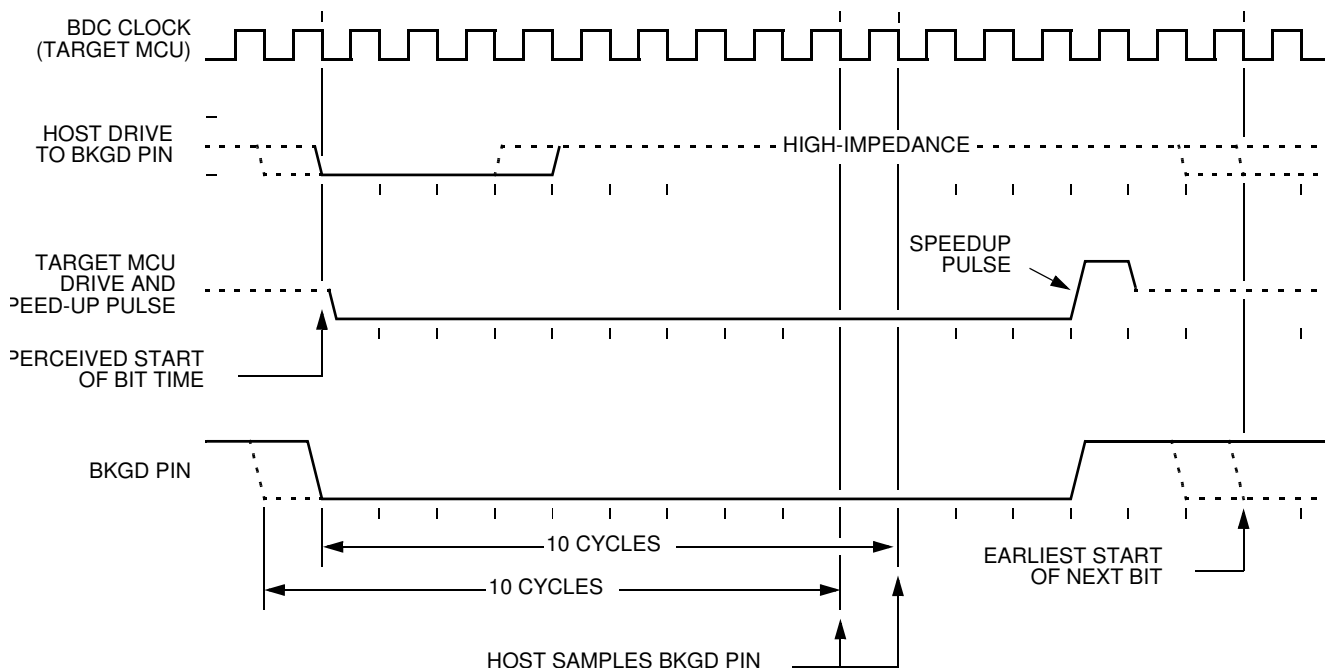
Figure 15-4 BDM Target-to-Host Serial Bit Timing (Logic 0)

15.2.3   BDC Commands

BDC commands are sent serially from a host computer to the BKGD pin of the target HCS08 MCU. All commands and data are sent MSB-first using a custom BDC communications protocol. Active background mode commands require that the target MCU is currently in the active background mode while non-intrusive commands may be issued at any time whether the target MCU is in active background mode or running a user application program.

Table 15-1 shows all HCS08 BDC commands, a shorthand description of their coding structure, and the meaning of each command.

Coding Structure Nomenclature

This nomenclature is used in Table 15-1 to describe the coding structure of the BDC commands.

Commands begin with an 8-bit hexadecimal command code in the host-to-target direction (most significant bit first)

/	=	separates parts of the command
d	=	delay 16 target BDC clock cycles
AAAA	=	a 16-bit address in the host-to-target direction
RD	=	8 bits of read data in the target-to-host direction
WD	=	8 bits of write data in the host-to-target direction
RD16	=	16 bits of read data in the target-to-host direction
WD16	=	16 bits of write data in the host-to-target direction
SS	=	the contents of BDCSCR in the target-to-host direction (STATUS)
CC	=	8 bits of write data for BDCSCR in the host-to-target direction (CONTROL)
RBKP	=	16 bits of read data in the target-to-host direction (from BDCBKPT breakpoint register)
WBKP	=	16 bits of write data in the host-to-target direction (for BDCBKPT breakpoint register)

Table 15-1 BDC Command Summary

Command Mnemonic	Active BDM/ Non-intrusive	Coding Structure	Description
SYNC	Non-intrusive	n/a[1]	Request a timed reference pulse to determine target BDC communication speed
ACK_ENABLE	Non-intrusive	D5/d	Enable acknowledge protocol. Refer to Freescale document order no. HCS08RMv1/D.
ACK_DISABLE	Non-intrusive	D6/d	Disable acknowledge protocol. Refer to Freescale document order no. HCS08RMv1/D.
BACKGROUND	Non-intrusive	90/d	Enter active background mode if enabled (ignore if ENBDM bit equals 0)
READ_STATUS	Non-intrusive	E4/SS	Read BDC status from BDCSCR
WRITE_CONTROL	Non-intrusive	C4/CC	Write BDC controls in BDCSCR
READ_BYTE	Non-intrusive	E0/AAAA/d/RD	Read a byte from target memory
READ_BYTE_WS	Non-intrusive	E1/AAAA/d/SS/RD	Read a byte and report status
READ_LAST	Non-intrusive	E8/SS/RD	Re-read byte from address just read and report status
WRITE_BYTE	Non-intrusive	C0/AAAA/WD/d	Write a byte to target memory
WRITE_BYTE_WS	Non-intrusive	C1/AAAA/WD/d/SS	Write a byte and report status
READ_BKPT	Non-intrusive	E2/RBKP	Read BDCBKPT breakpoint register
WRITE_BKPT	Non-intrusive	C2/WBKP	Write BDCBKPT breakpoint register
GO	Active BDM	08/d	Go to execute the user application program starting at the address currently in the PC
TRACE1	Active BDM	10/d	Trace 1 user instruction at the address in the PC, then return to active background mode
TAGGO	Active BDM	18/d	Same as GO but enable external tagging (HCS08 devices have no external tagging pin)
READ_A	Active BDM	68/d/RD	Read accumulator (A)
READ_CCR	Active BDM	69/d/RD	Read condition code register (CCR)
READ_PC	Active BDM	6B/d/RD16	Read program counter (PC)
READ_HX	Active BDM	6C/d/RD16	Read H and X register pair (H:X)
READ_SP	Active BDM	6F/d/RD16	Read stack pointer (SP)
READ_NEXT	Active BDM	70/d/RD	Increment H:X by one then read memory byte located at H:X
READ_NEXT_WS	Active BDM	71/d/SS/RD	Increment H:X by one then read memory byte located at H:X. Report status and data.
WRITE_A	Active BDM	48/WD/d	Write accumulator (A)
WRITE_CCR	Active BDM	49/WD/d	Write condition code register (CCR)
WRITE_PC	Active BDM	4B/WD16/d	Write program counter (PC)
WRITE_HX	Active BDM	4C/WD16/d	Write H and X register pair (H:X)
WRITE_SP	Active BDM	4F/WD16/d	Write stack pointer (SP)
WRITE_NEXT	Active BDM	50/WD/d	Increment H:X by one, then write memory byte located at H:X
WRITE_NEXT_WS	Active BDM	51/WD/d/SS	Increment H:X by one, then write memory byte located at H:X. Also report status.

NOTES:
1. The SYNC command is a special operation that does not have a command code.

The SYNC command is unlike other BDC commands because the host does not necessarily know the correct communications speed to use for BDC communications until after it has analyzed the response to the SYNC command.

To issue a SYNC command, the host:

- Drives the BKGD pin low for at least 128 cycles of the slowest possible BDC clock (The slowest clock is normally the reference oscillator/64 or the self-clocked rate/64.)
- Drives BKGD high for a brief speedup pulse to get a fast rise time (This speedup pulse is typically one cycle of the fastest clock in the system.)
- Removes all drive to the BKGD pin so it reverts to high impedance
- Monitors the BKGD pin for the sync response pulse

The target, upon detecting the SYNC request from the host (which is a much longer low time than would ever occur during normal BDC communications):

- Waits for BKGD to return to a logic high
- Delays 16 cycles to allow the host to stop driving the high speedup pulse
- Drives BKGD low for 128 BDC clock cycles
- Drives a 1-cycle high speedup pulse to force a fast rise time on BKGD
- Removes all drive to the BKGD pin so it reverts to high impedance

The host measures the low time of this 128-cycle sync response pulse and determines the correct speed for subsequent BDC communications. Typically, the host can determine the correct communication speed within a few percent of the actual target speed and the communication protocol can easily tolerate speed errors of several percent.

15.2.4   BDC Hardware Breakpoint

The BDC includes one relatively simple hardware breakpoint that compares the CPU address bus to a 16-bit match value in the BDCBKPT register. This breakpoint can generate a forced breakpoint or a tagged breakpoint. A forced breakpoint causes the CPU to enter active background mode at the first instruction boundary following any access to the breakpoint address. The tagged breakpoint causes the instruction opcode at the breakpoint address to be tagged so that the CPU will enter active background mode rather than executing that instruction if and when it reaches the end of the instruction queue. This implies that tagged breakpoints can only be placed at the address of an instruction opcode while forced breakpoints can be set at any address.

The breakpoint enable (BKPTEN) control bit in the BDC status and control register (BDCSCR) is used to enable the breakpoint logic (BKPTEN = 1). When BKPTEN = 0, its default value after reset, the breakpoint logic is disabled and no BDC breakpoints are requested regardless of the values in other BDC breakpoint registers and control bits. The force/tag select (FTS) control bit in BDCSCR is used to select forced (FTS = 1) or tagged (FTS = 0) type breakpoints.

The on-chip debug module (DBG) includes circuitry for two additional hardware breakpoints that are more flexible than the simple breakpoint in the BDC module.

15.3   On-Chip Debug System (DBG)

Because HCS08 devices do not have external address and data buses, the most important functions of an in-circuit emulator have been built onto the chip with the MCU. The debug system consists of an 8-stage FIFO that can store address or data bus information, and a flexible trigger system to decide when to capture bus information and what information to capture. The system relies on the single-wire background debug system to access debug control registers and to read results out of the eight stage FIFO.

The debug module includes control and status registers that are accessible in the user's memory map. These registers are located in the high register space to avoid using valuable direct page memory space.

Most of the debug module's functions are used during development, and user programs rarely access any of the control and status registers for the debug module. The one exception is that the debug system can provide the means to implement a form of ROM patching. This topic is discussed in greater detail in Section 15.3.6.

15.3.1   Comparators A and B

Two 16-bit comparators (A and B) can optionally be qualified with the R/W signal and an opcode tracking circuit. Separate control bits allow you to ignore R/W for each comparator. The opcode tracking circuitry optionally allows you to specify that a trigger will occur only if the opcode at the specified address is actually executed as opposed to only being read from memory into the instruction queue. The comparators are also capable of magnitude comparisons to support the inside range and outside range trigger modes. Comparators are disabled temporarily during all BDC accesses.

The A comparator is always associated with the 16-bit CPU address. The B comparator compares to the CPU address or the 8-bit CPU data bus, depending on the trigger mode selected. Because the CPU data bus is separated into a read data bus and a write data bus, the RWAEN and RWA control bits have an additional purpose, in full address plus data comparisons they are used to decide which of these buses to use in the comparator B data bus comparisons. If RWAEN = 1 (enabled) and RWA = 0 (write), the CPU's write data bus is used. Otherwise, the CPU's read data bus is used.

The currently selected trigger mode determines what the debugger logic does when a comparator detects a qualified match condition. A match can cause:

- Generation of a breakpoint to the CPU
- Storage of data bus values into the FIFO
- Starting to store change-of-flow addresses into the FIFO (begin type trace)
- Stopping the storage of change-of-flow addresses into the FIFO (end type trace)

15.3.2   Bus Capture Information and FIFO Operation

The usual way to use the FIFO is to setup the trigger mode and other control options, then arm the debugger. When the FIFO has filled or the debugger has stopped storing data into the FIFO, you would read the information out of it in the order it was stored into the FIFO. Status bits indicate the number of words of valid information that are in the FIFO as data is stored into it. If a trace run is manually halted by writing 0 to ARM before the FIFO is full (CNT = 1:0:0:0), the information is shifted by one position and the host must perform ((8 − CNT) − 1) dummy reads of the FIFO to advance it to the first significant entry in the FIFO.

In most trigger modes, the information stored in the FIFO consists of 16-bit change-of-flow addresses. In these cases, read DBGFH then DBGFL to get one coherent word of information out of the FIFO. Reading DBGFL (the low-order byte of the FIFO data port) causes the FIFO to shift so the next word of information is available at the FIFO data port. In the event-only trigger modes (see Section 15.3.5), 8-bit data information is stored into the FIFO. In these cases, the high-order half of the FIFO (DBGFH) is not used and data is read out of the FIFO by simply reading DBGFL. Each time DBGFL is read, the FIFO is shifted so the next data value is available through the FIFO data port at DBGFL.

In trigger modes where the FIFO is storing change-of-flow addresses, there is a delay between CPU addresses and the input side of the FIFO. Because of this delay, if the trigger event itself is a change-of-flow address or a change-of-flow address appears during the next two bus cycles after a trigger event starts the FIFO, it will not be saved into the FIFO. In the case of an end-trace, if the trigger event is a change-of-flow, it will be saved as the last change-of-flow entry for that debug run.

The FIFO can also be used to generate a profile of executed instruction addresses when the debugger is not armed. When ARM = 0, reading DBGFL causes the address of the most-recently fetched opcode to be saved in the FIFO. To use the profiling feature, a host debugger would read addresses out of the FIFO by reading DBGFH then DBGFL at regular periodic intervals. The first eight values would be discarded because they correspond to the eight DBGFL reads needed to initially fill the FIFO. Additional periodic reads of DBGFH and DBGFL return delayed information about executed instructions so the host debugger can develop a profile of executed instruction addresses.

15.3.3   Change-of-Flow Information

To minimize the amount of information stored in the FIFO, only information related to instructions that cause a change to the normal sequential execution of instructions is stored. With knowledge of the source and object code program stored in the target system, an external debugger system can reconstruct the path of execution through many instructions from the change-of-flow information stored in the FIFO.

For conditional branch instructions where the branch is taken (branch condition was true), the source address is stored (the address of the conditional branch opcode). Because BRA and BRN instructions are not conditional, these events do not cause change-of-flow information to be stored in the FIFO.

Indirect JMP and JSR instructions use the current contents of the H:X index register pair to determine the destination address, so the debug system stores the run-time destination address for any indirect JMP or JSR. For interrupts, RTI, or RTS, the destination address is stored in the FIFO as change-of-flow information.

15.3.4   Tag vs. Force Breakpoints and Triggers

Tagging is a term that refers to identifying an instruction opcode as it is fetched into the instruction queue, but not taking any other action until and unless that instruction is actually executed by the CPU. This distinction is important because any change-of-flow from a jump, branch, subroutine call, or interrupt causes some instructions that have been fetched into the instruction queue to be thrown away without being executed.

A force-type breakpoint waits for the current instruction to finish and then acts upon the breakpoint request. The usual action in response to a breakpoint is to go to active background mode rather than continuing to the next instruction in the user application program.

The tag vs. force terminology is used in two contexts within the debug module. The first context refers to breakpoint requests from the debug module to the CPU. The second refers to match signals from the comparators to the debugger control logic. When a tag-type break request is sent to the CPU, a signal is entered into the instruction queue along with the opcode so that if/when this opcode ever executes, the CPU will effectively replace the tagged opcode with a BGND opcode so the CPU goes to active background mode rather than executing the tagged instruction. When the TRGSEL control bit in the DBGT register is set to select tag-type operation, the output from comparator A or B is qualified by a block of logic in the debug module that tracks opcodes and only produces a trigger to the debugger if the opcode at the compare address is actually executed. There is separate opcode tracking logic for each comparator so more than one compare event can be tracked through the instruction queue at a time.

15.3.5   Trigger Modes

The trigger mode controls the overall behavior of a debug run. The 4-bit TRG field in the DBGT register selects one of nine trigger modes. When TRGSEL = 1 in the DBGT register, the output of the comparator must propagate through an opcode tracking circuit before triggering FIFO actions. The BEGIN bit in DBGT chooses whether the FIFO begins storing data when the qualified trigger is detected (begin trace), or the FIFO stores data in a circular fashion from the time it is armed until the qualified trigger is detected (end trigger).

A debug run is started by writing a 1 to the ARM bit in the DBGC register, which sets the ARMF flag and clears the AF and BF flags and the CNT bits in DBGS. A begin-trace debug run ends when the FIFO gets full. An end-trace run ends when the selected trigger event occurs. Any debug run can be stopped manually by writing a 0 to ARM or DBGEN in DBGC.

In all trigger modes except event-only modes, the FIFO stores change-of-flow addresses. In event-only trigger modes, the FIFO stores data in the low-order eight bits of the FIFO.

The BEGIN control bit is ignored in event-only trigger modes and all such debug runs are begin type traces. When TRGSEL = 1 to select opcode fetch triggers, it is not necessary to use R/W in comparisons because opcode tags would only apply to opcode fetches that are always read cycles. It would also be unusual to specify TRGSEL = 1 while using a full mode trigger because the opcode value is normally known at a particular address.

The following trigger mode descriptions only state the primary comparator conditions that lead to a trigger. Either comparator can usually be further qualified with R/W by setting RWAEN (RWBEN) and the corresponding RWA (RWB) value to be matched against R/W. The signal from the comparator with optional R/W qualification is used to request a CPU breakpoint if BRKEN = 1 and TAG determines whether the CPU request will be a tag request or a force request.

A-Only — Trigger when the address matches the value in comparator A

A OR B — Trigger when the address matches either the value in comparator A or the value in comparator B

A Then B — Trigger when the address matches the value in comparator B but only after the address for another cycle matched the value in comparator A. There can be any number of cycles after the A match and before the B match.

A AND B Data (Full Mode) — This is called a full mode because address, data, and R/W (optionally) must match within the same bus cycle to cause a trigger event. Comparator A checks address, the low byte of comparator B checks data, and R/W is checked against RWA if RWAEN = 1. The high-order half of comparator B is not used.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

A AND NOT B Data (Full Mode) — Address must match comparator A, data must not match the low half of comparator B, and R/W must match RWA if RWAEN = 1. All three conditions must be met within the same bus cycle to cause a trigger.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

Event-Only B (Store Data) — Trigger events occur each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

A Then Event-Only B (Store Data) — After the address has matched the value in comparator A, a trigger event occurs each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

Inside Range (A ≤ Address ≤ B) — A trigger occurs when the address is greater than or equal to the value in comparator A and less than or equal to the value in comparator B at the same time.

Outside Range (Address < A or Address > B) — A trigger occurs when the address is either less than the value in comparator A or greater than the value in comparator B.

15.3.6   Hardware Breakpoints

The BRKEN control bit in the DBGC register may be set to 1 to allow any of the trigger conditions described in to be used to generate a hardware breakpoint request to the CPU. TAG in DBGC controls whether the breakpoint request will be treated as a tag-type breakpoint or a force-type breakpoint. A tag breakpoint causes the current opcode to be marked as it enters the instruction queue. If a tagged opcode reaches the end of the pipe, the CPU executes a BGND instruction to go to active background mode rather than executing the tagged opcode. A force-type breakpoint causes the CPU to finish the current instruction and then go to active background mode.

If the background mode has not been enabled (ENBDM = 1) by a serial WRITE_CONTROL command through the BKGD pin, the CPU will execute an SWI instruction instead of going to active background mode.

15.4   Register Definition

This section contains the descriptions of the BDC and DBG registers and control bits.

Refer to the high-page register summary in the device overview chapter of this data sheet for the absolute address assignments for all DBG registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

15.4.1   BDC Registers and Control Bits

The BDC has two registers:
   •   The BDC status and control register (BDCSCR) is an 8-bit register containing control and status bits for the background debug controller.
   •   The BDC breakpoint match register (BDCBKPT) holds a 16-bit breakpoint match address.

These registers are accessed with dedicated serial BDC commands and are not located in the memory space of the target MCU (so they do not have addresses and cannot be accessed by user programs).

Some of the bits in the BDCSCR have write limitations; otherwise, these registers may be read or written at any time. For example, the ENBDM control bit may not be written while the MCU is in active background mode. (This prevents the ambiguous condition of the control bit forbidding active background mode while the MCU is already in active background mode.) Also, the four status bits (BDMACT, WS, WSF, and DVF) are read-only status indicators and can never be written by the WRITE_CONTROL serial BDC command. The clock switch (CLKSW) control bit may be read or written at any time.

15.4.1.1   BDC Status and Control Register (BDCSCR)

This register can be read or written by serial BDC commands (READ_STATUS and WRITE_CONTROL) but is not accessible to user programs because it is not located in the normal memory map of the MCU.

	7	6	5	4	3	2	1	0
R	ENBDM	BDMACT	BKPTEN	FTS	CLKSW	WS	WSF	DVF
W								
Normal Reset	0	0	0	0	0	0	0	0
Reset in Active BDM:	1	1	0	0	1	0	0	0

　　　= Unimplemented or Reserved

Figure 15-5   BDC Status and Control Register (BDCSCR)

Table 15-2 BDCSCR Register Field Descriptions

Field	Description
7 ENBDM	**Enable BDM (Permit Active Background Mode)** — Typically, this bit is written to 1 by the debug host shortly after the beginning of a debug session or whenever the debug host resets the target and remains 1 until a normal reset clears it. 0 BDM cannot be made active (non-intrusive commands still allowed) 1 BDM can be made active to allow active background mode commands
6 BDMACT	**Background Mode Active Status** — This is a read-only status bit. 0 BDM not active (user application program running) 1 BDM active and waiting for serial commands
5 BKPTEN	**BDC Breakpoint Enable** — If this bit is clear, the BDC breakpoint is disabled and the FTS (force tag select) control bit and BDCBKPT match register are ignored. 0 BDC breakpoint disabled 1 BDC breakpoint enabled
4 FTS	**Force/Tag Select** — When FTS = 1, a breakpoint is requested whenever the CPU address bus matches the BDCBKPT match register. When FTS = 0, a match between the CPU address bus and the BDCBKPT register causes the fetched opcode to be tagged. If this tagged opcode ever reaches the end of the instruction queue, the CPU enters active background mode rather than executing the tagged opcode. 0 Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction 1 Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode)
3 CLKSW	**Select Source for BDC Communications Clock** — CLKSW defaults to 0, which selects the alternate BDC clock source. 0 Alternate BDC clock source 1 MCU bus clock
2 WS	**Wait or Stop Status** — When the target CPU is in wait or stop mode, most BDC commands cannot function. However, the BACKGROUND command can be used to force the target CPU out of wait or stop and into active background mode where all BDC commands work. Whenever the host forces the target MCU into active background mode, the host should issue a READ_STATUS command to check that BDMACT = 1 before attempting other BDC commands. 0 Target CPU is running user application code or in active background mode (was not in wait or stop mode when background became active) 1 Target CPU is in wait or stop mode, or a BACKGROUND command was used to change from wait or stop to active background mode
1 WSF	**Wait or Stop Failure Status** — This status bit is set if a memory access command failed due to the target CPU executing a wait or stop instruction at or about the same time. The usual recovery strategy is to issue a BACKGROUND command to get out of wait or stop mode into active background mode, repeat the command that failed, then return to the user program. (Typically, the host would restore CPU registers and stack values and re-execute the wait or stop instruction.) 0 Memory access did not conflict with a wait or stop instruction 1 Memory access command failed because the CPU entered wait or stop mode
0 DVF	**Data Valid Failure Status** — This status bit is not used in the MPXY8300 Series because it does not have any slow access memory. 0 Memory access did not conflict with a slow memory access 1 Memory access command failed because CPU was not finished with a slow memory access

15.4.1.2 BDC Breakpoint Match Register (BDCBKPT)

This 16-bit register holds the address for the hardware breakpoint in the BDC. The BKPTEN and FTS control bits in BDCSCR are used to enable and configure the breakpoint logic. Dedicated serial BDC commands (READ_BKPT and WRITE_BKPT) are used to read and write the BDCBKPT register but is not accessible to user programs because it is not located in the normal memory map of the MCU. Breakpoints are normally set while the target MCU is in active background mode before running the user application program. For additional information about setup and use of the hardware breakpoint logic in the BDC, refer to Section 15.2.4.

MPXY8300 Series

15.4.2 System Background Debug Force Reset Register (SBDFR)

This register contains a single write-only control bit. A serial background mode command such as WRITE_BYTE must be used to write to SBDFR. Attempts to write this register from a user program are ignored. Reads always return 0x00.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								BDFR[1]
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

NOTES:
1. BDFR is writable only through serial background mode debug commands, not from user programs.

Figure 15-6 System Background Debug Force Reset Register (SBDFR)

Table 15-3 SBDFR Register Field Description

Field	Description
0 BDFR	**Background Debug Force Reset** — A serial active background mode command such as WRITE_BYTE allows an external debug host to force a target system reset. Writing 1 to this bit forces an MCU reset. This bit cannot be written from a user program.

15.4.3 DBG Registers and Control Bits

The debug module includes nine bytes of register space for three 16-bit registers and three 8-bit control and status registers. These registers are located in the high register space of the normal memory map so they are accessible to normal application programs. These registers are rarely if ever accessed by normal user application programs with the possible exception of a ROM patching mechanism that uses the breakpoint logic.

15.4.3.1 Debug Comparator A High Register (DBGCAH)

This register contains compare value bits for the high-order eight bits of comparator A. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

15.4.3.2 Debug Comparator A Low Register (DBGCAL)

This register contains compare value bits for the low-order eight bits of comparator A. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

15.4.3.3 Debug Comparator B High Register (DBGCBH)

This register contains compare value bits for the high-order eight bits of comparator B. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

15.4.3.4 Debug Comparator B Low Register (DBGCBL)

This register contains compare value bits for the low-order eight bits of comparator B. This register is forced to 0x00 at reset and can be read at any time or written at any time unless ARM = 1.

15.4.3.5 Debug FIFO High Register (DBGFH)

This register provides read-only access to the high-order eight bits of the FIFO. Writes to this register have no meaning or effect. In the event-only trigger modes, the FIFO only stores data into the low-order byte of each FIFO word, so this register is not used and will read 0x00.

Reading DBGFH does not cause the FIFO to shift to the next word. When reading 16-bit words out of the FIFO, read DBGFH before reading DBGFL because reading DBGFL causes the FIFO to advance to the next word of information.

15.4.3.6 Debug FIFO Low Register (DBGFL)

This register provides read-only access to the low-order eight bits of the FIFO. Writes to this register have no meaning or effect.

Reading DBGFL causes the FIFO to shift to the next available word of information. When the debug module is operating in event-only modes, only 8-bit data is stored into the FIFO (high-order half of each FIFO word is unused). When reading 8-bit words out of the FIFO, simply read DBGFL repeatedly to get successive bytes of data from the FIFO. It isn't necessary to read DBGFH in this case.

Do not attempt to read data from the FIFO while it is still armed (after arming but before the FIFO is filled or ARMF is cleared) because the FIFO is prevented from advancing during reads of DBGFL. This can interfere with normal sequencing of reads from the FIFO.

Reading DBGFL while the debugger is not armed causes the address of the most-recently fetched opcode to be stored to the last location in the FIFO. By reading DBGFH then DBGFL periodically, external host software can develop a profile of program execution. After eight reads from the FIFO, the ninth read will return the information that was stored as a result of the first read. To use the profiling feature, read the FIFO eight times without using the data to prime the sequence and then begin using the data to get a delayed picture of what addresses were being executed. The information stored into the FIFO on reads of DBGFL (while the FIFO is not armed) is the address of the most-recently fetched opcode.

15.4.3.7 Debug Control Register (DBGC)

This register can be read or written at any time.

	7	6	5	4	3	2	1	0
R	DBGEN	ARM	TAG	BRKEN	RWA	RWAEN	RWB	RWBEN
W								
Reset	0	0	0	0	0	0	0	0

Figure 15-7 Debug Control Register (DBGC)

Table 15-4 DBGC Register Field Descriptions

Field	Description
7 DBGEN	**Debug Module Enable** — Used to enable the debug module. DBGEN cannot be set to 1 if the MCU is secure. 0 DBG disabled 1 DBG enabled
6 ARM	**Arm Control** — Controls whether the debugger is comparing and storing information in the FIFO. A write is used to set this bit (and ARMF) and completion of a debug run automatically clears it. Any debug run can be manually stopped by writing 0 to ARM or to DBGEN. 0 Debugger not armed 1 Debugger armed
5 TAG	**Tag/Force Select** — Controls whether break requests to the CPU will be tag or force type requests. If BRKEN = 0, this bit has no meaning or effect. 0 CPU breaks requested as force type requests 1 CPU breaks requested as tag type requests
4 BRKEN	**Break Enable** — Controls whether a trigger event will generate a break request to the CPU. Trigger events can cause information to be stored in the FIFO without generating a break request to the CPU. For an end trace, CPU break requests are issued to the CPU when the comparator(s) and R/W meet the trigger requirements. For a begin trace, CPU break requests are issued when the FIFO becomes full. TRGSEL does not affect the timing of CPU break requests. 0 CPU break requests not enabled 1 Triggers cause a break request to the CPU

Table 15-4 DBGC Register Field Descriptions (Continued)

Field	Description
3 RWA	**R/W Comparison Value for Comparator A** — When RWAEN = 1, this bit determines whether a read or a write access qualifies comparator A. When RWAEN = 0, RWA and the R/W signal do not affect comparator A. 0 Comparator A can only match on a write cycle 1 Comparator A can only match on a read cycle
2 RWAEN	**Enable R/W for Comparator A** — Controls whether the level of R/W is considered for a comparator A match. 0 R/W is not used in comparison A 1 R/W is used in comparison A
1 RWB	**R/W Comparison Value for Comparator B** — When RWBEN = 1, this bit determines whether a read or a write access qualifies comparator B. When RWBEN = 0, RWB and the R/W signal do not affect comparator B. 0 Comparator B can match only on a write cycle 1 Comparator B can match only on a read cycle
0 RWBEN	**Enable R/W for Comparator B** — Controls whether the level of R/W is considered for a comparator B match. 0 R/W is not used in comparison B 1 R/W is used in comparison B

15.4.3.8 Debug Trigger Register (DBGT)

This register can be read any time, but may be written only if ARM = 0, except bits 4 and 5 are hard-wired to 0s.

	7	6	5	4	3	2	1	0
R	TRGSEL	BEGIN	0	0	TRG3	TRG2	TRG1	TRG0
W								
Reset	0	0	0	0	0	0	0	0

[ ] = Unimplemented or Reserved
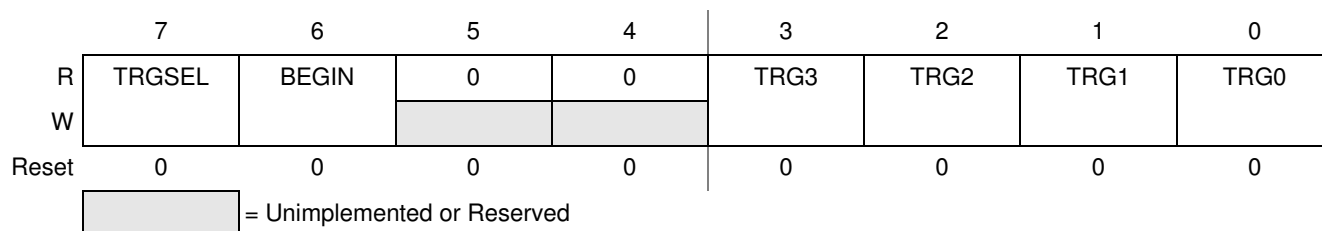
Figure 15-8 Debug Trigger Register (DBGT)

Table 15-5 DBGT Register Field Descriptions

Field	Description
7 TRGSEL	**Trigger Type** — Controls whether the match outputs from comparators A and B are qualified with the opcode tracking logic in the debug module. If TRGSEL is set, a match signal from comparator A or B must propagate through the opcode tracking logic and a trigger event is only signalled to the FIFO logic if the opcode at the match address is actually executed. 0 Trigger on access to compare address (force) 1 Trigger if opcode at compare address is executed (tag)
6 BEGIN	**Begin/End Trigger Select** — Controls whether the FIFO starts filling at a trigger or fills in a circular manner until a trigger ends the capture of information. In event-only trigger modes, this bit is ignored and all debug runs are assumed to be begin traces. 0 Data stored in FIFO until trigger (end trace) 1 Trigger initiates data storage (begin trace)
3:0 TRG[3:0]-+-	**Select Trigger Mode** — Selects one of nine triggering modes, as described below. 0000A-only 0001 A OR B 0010A Then B 0011Event-only B (store data) 0100A then event-only B (store data) 0101A AND B data (full mode) 0110A AND NOT B data (full mode) 0111Inside range: A ≤ address ≤ B 1000Outside range: address < A or address > B 1001 – 1111 (No trigger)

15.4.3.9 Debug Status Register (DBGS)

This is a read-only status register.

	7	6	5	4	3	2	1	0
R	AF	BF	ARMF	0	CNT3	CNT2	CNT1	CNT0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-9 Debug Status Register (DBGS)

Table 15-6 DBGS Register Field Descriptions

Field	Description
7 AF	**Trigger Match A Flag** — AF is cleared at the start of a debug run and indicates whether a trigger match A condition was met since arming. 0 Comparator A has not matched 1 Comparator A match
6 BF	**Trigger Match B Flag** — BF is cleared at the start of a debug run and indicates whether a trigger match B condition was met since arming. 0 Comparator B has not matched 1 Comparator B match
5 ARMF	**Arm Flag** — While DBGEN = 1, this status bit is a read-only image of ARM in DBGC. This bit is set by writing 1 to the ARM control bit in DBGC (while DBGEN = 1) and is automatically cleared at the end of a debug run. A debug run is completed when the FIFO is full (begin trace) or when a trigger event is detected (end trace). A debug run can also be ended manually by writing 0 to ARM or DBGEN in DBGC. 0 Debugger not armed 1 Debugger armed
3:0 CNT[3:0]	**FIFO Valid Count** — These bits are cleared at the start of a debug run and indicate the number of words of valid data in the FIFO at the end of a debug run. The value in CNT does not decrement as data is read out of the FIFO. The external debug host is responsible for keeping track of the count as information is read out of the FIFO. 0000Number of valid words in FIFO = No valid data 0001Number of valid words in FIFO = 1 0010Number of valid words in FIFO = 2 0011Number of valid words in FIFO = 3 0100Number of valid words in FIFO = 4 0101Number of valid words in FIFO = 5 0110Number of valid words in FIFO = 6 0111Number of valid words in FIFO = 7 1000Number of valid words in FIFO = 8

MPXY8300 Series

SECTION 16 BATTERY CHARGE CONSUMPTION MODELING

The supply current consumed by the MPXY8300 Series can be estimated using the following basic model.

16.1 Standby Current

The overall charge consumed by the standby features is:

$$Q_{STDBY} = t_{TOT} \times \frac{(I_{STDBY} + I_{LF})}{1000}$$

where:

Q_{STDBY} = Standby charge over lifetime, t_{TOT}, in mA-hr

t_{TOT} = Total lifetime in hours

I_{STDBY} = General standby current in μA

I_{LF} = LFR detector (if used) current in μA

16.2 Measurement Events

The overall charge consumed by the measurements is:

$$Q_{MEAS} = \frac{1}{1000} \times (n_{PRESS} \times Q_{PRESS} + n_{TEMP} \times Q_{TEMP} + n_{VOLT} \times Q_{VOLT})$$

where:

Q_{MEAS} = Total measurement charge over lifetime in mA-sec

Q_{PRESS} = Measurement charge per pressure measurement in μA-sec

Q_{TEMP} = Measurement charge per temperature measurement in μA-sec

Q_{VOLT} = Measurement charge per voltage measurement in μA-sec

n_{PRESS} = Total number of pressure measurements over lifetime

n_{TEMP} = Total number of temperature measurements over lifetime

n_{VOLT} = Total number of voltage measurements over lifetime

16.3 Transmission Events

The overall charge consumed by the transmissions is:

$$Q_{XMT} = \frac{Q_{FRM}}{1000} \times F \times n_{XMT}$$

where:

Q_{XMT} = Transmit charge over lifetime, t_{TOT}, in mA-hr

Q_{FRM} = Transmit charge per frame of data in μA-sec

n_{XMT} = Number of transmissions over lifetime

F = Frames transmitted during each datagram

16.4 Total Consumption

The overall charge consumed is:

$$Q_{TOT} = \frac{(Q_{STDBY} + Q_{MEAS} + Q_{XMT})}{(1 - Y \times SD/100)}$$

where:

Q_{TOT}	=	Total charge over lifetime, t_{TOT}, in mA-hr
Q_{STDBY}	=	Standby charge over lifetime in mA-hr
Q_{MEAS}	=	Measurement charge over lifetime in mA-hr
Q_{XMT}	=	Transmit charge over lifetime in mA-hr
Y	=	Lifetime in years
SD	=	Battery self-discharge rate in%/year

Additional margin in battery capacity can be added to the calculated value of Q_{TOT}.

SECTION 17   ELECTRICAL SPECIFICATIONS

17.1   Parameter Identification and Validation

Each parameter in this section is indicated by a numerical number given in the "#" column.

The electrical parameters shown in this section are validated by various methods as designated in the "V" column. To give the customer a better understanding, the classifications given in Table 17-1 are used and the parameters are tagged accordingly.

Table 17-1   Parameter Classifications

V	Description
F	These parameters are validated by final testing of the complete packaged device.
P	These parameters are validated by probe testing of the individual devices used in the packaged device.
C	These parameters are validated by characterization testing of the complete packaged device.
D	These parameters are validated by design intent and simulation.

17.2   Absolute Maximum Ratings

Maximum ratings are the extreme limits to which the device can be exposed without permanently damaging it. The device contains circuitry to protect the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in the table below. Keep V_{IN} and V_{OUT} within the range $V_{SS} \leq (V_{IN}$ or $V_{OUT}) \leq V_{DD}$.

#	V	Rating	Symbol	Value	Unit
1	C	Supply Voltage (V_{DD}, AV_{DD})	V_{DD}	-0.3 to +3.8	V
2	C	Input Voltage XI	V_{IN}	-0.3 to V_{DD}+0.3	V
3	C	PTA0, PTA1, PTA2, PTA3, PTA5, PTA6	V_{IN}	-0.3 to V_{DD}+0.3	V
4	C	BKGD	V_{IN}	-0.3 to V_{DD}+0.3	V
5	C	Input Current XI	I_{IN}	10	mA
6	C	PTA0, PTA1, PTA2, PTA3, PTA5, PTA6	I_{IN}	10	mA
7	C	BKGD	I_{IN}	10	mA

8	C	Substrate Current Injection
		    Current from any pin to V_{SS} - 0.3 VDC
		    XI, PTA0, PTA1, PTA2, PTA3, PTA5, PTA6,BKGD
			I_{SUB}	600	µA

9	C	PTA0, PTA1	I_{SUB}	2	mA
10	C	Latchup Current Current to/from any pin to supply rails ± 0.3 VDC	I_{LATCH}	± 100	mA
11	C	Electrostatic Discharge Human Body Model (HBM), all pins other than RF	V_{ESD}	± 2000	V
12	C	Human Body Model (HBM), RF pin	V_{ESD}	± 4000	V
13	C	Charged Device Model (CDM), Pins 1, 10, 11, 20	V_{ESD}	± 750	V
14	C	Pins 2-9, 12-19	V_{ESD}	± 500	V
15	C	Machine Model (MM)	V_{ESD}	± 200	V
16	C	Maximum Storage Temperature Range	T_{stg}	-40 to +150	°C

MPXY8300 Series

17.3 Operating Range

The limits normally expected in the application which define range of operation.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Operating Supply Voltage ($V_{DD} = AV_{DD}$)		V_L		V_H	
17	P	Measurements	V_{DD}	2.3	3.0	3.6	V
18	P	RF Transmissions and LFR Operation	V_{DD}	2.1	3.0	3.6	V
19	P	Charge Pump (T_A = -40°C to 25°C)	V_{DD}	2.1	—	2.7	V
		MCU operation (CPU, ADC10, RAM, TPM1)					
20	P	FLASH write (-40 to +125°C)	V_{DD}	2.3	3.0	3.6	V
21	P	FLASH write (-40 to +85°C)	V_{DD}	2.1	3.0	3.6	V
22	P	FLASH read	V_{DD}	1.8	3.0	3.6	V
		Operating Temperature Range		T_L		T_H	
23	C	Continuous Temperature Range, all modes	T_A	-40	—	+125	°C
24	C	Stop1 mode (<3 hours above 125°C and <45 minutes per excursion above 125°C)	T_A	-40	—	+150	°C

17.4 Electrical Characteristics

$2.1 \leq V_{DD} \leq 3.6$, $T_L \leq T_A \leq T_H$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
25	P	Output High Voltage (I_{Load} = 5 mA) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6	V_{OH}	V_{DD}-0.35	—	—	V
26	P	Output Low Voltage (I_{Load} = -5 mA) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6	V_{OL}	—	—	0.35	V
27	P	Output High Voltage V_{RF} (I_{Load} = 2 mA)	V_{OH}	V_{DD}-0.1	—	—	V
28	P	Output Low Voltage V_{RF} (I_{Load} = -1 mA)	V_{OL}	—	—	0.4	V
29	P	Input High Voltage ($2.3 < V_{DD} \leq V_H$) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6, BKGD	V_{IH}	0.7 x V_{DD}	—	—	V
30	P	Input High Voltage ($V_L \leq V_{DD} \leq 2.3$) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6, BKGD	V_{IH}	0.85 x V_{DD}	—	—	V
31	P	Input Low Voltage ($2.3 < V_{DD} \leq V_H$) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6, BKGD	V_{IL}	V_{SS}	—	0.35 x V_{DD}	V
32	P	Input Low Voltage ($V_L \leq V_{DD} \leq 2.3$) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6, BKGD	V_{IL}	V_{SS}	—	0.3 x V_{DD}	V
33	P	Input High Current (at V_{IH}) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6	I_{IH}	+4	—	+7	µA
34	P	BKGD	I_{IH}	-1	0	+1	µA
35	P	Input Low Current (at V_{IL}) PTA0, PTA1, PTA2, PTA3, PTA5, PTA6	I_{IH}	+4	—	+7	µA
36	P	BKGD	I_{IL}	-1	0	+1	µA

17.5 Power Consumption (MCU)

$2.3 \leq V_{DD} \leq 3.6$, $T_A = 0$ to $70\,°C$ unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Standby Supply Current (V_{DD} = 3V, T_A = 25 °C)					
37	P	Stop1 mode, LFR, LVD and TR all off	I_{STDBY}	—	0.36	0.9	μA
38	P	Adder for LFR (continuous ON, any mode)	I_{STDBY}	—	93	112	μA
39	P	Adder for temperature restart (TR)	I_{STDBY}	—	10	20	μA
40	P	Stop4 mode, LFR, LVD and TR all off	I_{STDBY}	—	73	95	μA
		Standby Supply Current (V_{DD} = 3V, T_A = 125 °C)					
41	C	Stop1 mode, LFR, LVD and TR all off	I_{STDBY}	—	9	19.5	μA
42	C	Adder for LFR (continuous ON, any mode)	I_{STDBY}	—	80	112	μA
43	C	Adder for temperature restart (TR)	I_{STDBY}	—	10	20	μA
44	C	Stop4 mode, LFR, LVD and TR all off	I_{STDBY}	—	95	140	μA
		MCU Operate Current (V_{DD} = 3V, T_A = 25 °C)					
45	C	0.5 MHz f_{BUS}, BUSCLKS1 = 1, BUSCLKS0 = 1	I_{DD}	—	0.68	0.8	mA
46	P	1 MHz f_{BUS}, BUSCLKS1 = 1, BUSCLKS0 = 0	I_{DD}	—	0.94	1.1	mA
47	C	2 MHz f_{BUS}, BUSCLKS1 = 0, BUSCLKS0 = 1	I_{DD}	—	1.46	1.7	mA
48	C	4 MHz f_{BUS}, BUSCLKS1 = 0, BUSCLKS0 = 0	I_{DD}	—	2.50	2.9	mA

17.6 Power Consumption (Measurements)

$2.3 \leq V_{DD} \leq 3.3$, $T_A = -40$ to $125\,°C$ unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Pressure and Temperature Measurement [1]					
49	C	Sensor measurement time [2]	t_{PM}	—	3.3	—	mSec
50	C	Peak current (V_{DD} = 3.3V) [3]	I_P	—	4.0	—	mA
51	C	Total power consumption	Q_P	—	6.28	7.3	μA-sec
		Acceleration Measurement (X- or Z-axis) [4]					
52	C	Sensor measurement time (LP Filter ON) [2]	t_{AM}	—	3.82	—	mSec
53	C	Peak current (V_{DD} = 3.3V) [3]	IA	—	3.4	—	mA
54	C	Total power consumption (LP Filter ON)	Q_A	—	3.77	4.5	μA-sec
		Temperature Measurement					
55	C	Sensor measurement time [2]	t_{TM}	—	1.13	—	mSec
56	C	Peak current (V_{DD} = 3.3V) [3]	IT	—	3.4	—	mA
57	C	Total power consumption	Q_T	—	1.17	1.4	μA-sec
		Voltage Measurement					
58	C	Sensor measurement time [2]	t_{VM}	—	0.26	—	mSec
59	C	Peak current (V_{DD} = 3.3V) [3]	IV	—	3.4	—	mA
60	C	Total power consumption	Q_V	—	0.35	0.8	μA-sec

NOTES:
1. Fully compensated pressure and temperature reading using the PCOMP firmware routine with average of 8 readings. Power consumption can be reduced if readings are uncompensated.
2. Measurement times dependent on clock tolerances.
3. Peak currents measured using external network shown in Figure 17-5 with R_{BATT} equal to zero ohms.
4. Fully compensated acceleration reading using the ACOMP firmware routine with single reading and the 500 Hz low-pass filter active. Power consumption can be reduced if readings are uncompensated.

17.7 Power Consumption (RF Transmissions, 315 MHz Carrier Frequency)

$2.1 \leq V_{DD} \leq 3.6$, $T_A = -40$ to $125\,°C$ unless otherwise specified.

Target power output 5 dBm using Dynamic RF Power Correction firmware routine per Section 14

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		RF Transmission Supply Current, T_A = -40 °C					
		V_{DD} = 2.1V, PWR4:0 = 01110					
61	C	Data 1, FSK or ASK	I_{DD}	—	7.44	8.20	mA
62	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 2.5V, PWR4:0 = 01110					
63	C	Data 1, FSK or ASK	I_{DD}	—	7.95	8.75	mA
64	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 3.0V, PWR4:0 = 01101					
65	P	Data 1, FSK or ASK	I_{DD}	—	8.21	9.00	mA
66	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 3.3V, PWR4:0 = 01101					
67	C	Data 1, FSK or ASK	I_{DD}	—	8.48	9.30	mA
68	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		RF Transmission Supply Current, T_A = 25 °C					
		V_{DD} = 2.1V, PWR4:0 = 01110					
69	C	Data 1, FSK or ASK	I_{DD}	—	7.82	8.45	mA
70	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 2.5V, PWR4:0 = 01110					
71	C	Data 1, FSK or ASK	I_{DD}	—	8.25	8.90	mA
72	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 3.0V, PWR4:0 = 01011					
73	P	Data 1, FSK or ASK	I_{DD}	—	8.53	9.20	mA
74	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 3.3V, PWR4:0 = 01011					
75	C	Data 1, FSK or ASK	I_{DD}	—	8.80	9.50	mA
76	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		RF Transmission Supply Current, T_A = 125 °C					
		V_{DD} = 2.1V, PWR4:0 = 01110					
77	C	Data 1, FSK or ASK	I_{DD}	—	8.53	9.20	mA
78	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 2.5V, PWR4:0 = 01110					
79	C	Data 1, FSK or ASK	I_{DD}	—	9.07	9.70	mA
80	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 3.0V, PWR4:0 = 01011					
81	P	Data 1, FSK or ASK	I_{DD}	—	9.45	10.15	mA
82	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		V_{DD} = 3.3V, PWR4:0 = 01011					
83	C	Data 1, FSK or ASK	I_{DD}	—	9.75	10.45	mA
84	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	TBD	TBD	mA
		RF Transmission Supply Current					
85	C	Delta I_{DD} per PWR4:0 bit step	ΔI_{DD}	—	0.5	—	mA

17.8 Power Consumption (RF Transmissions, 434 MHz Carrier Frequency)

$2.1 \le V_{DD} \le 3.6$, $T_A = -40$ to $125\,°C$ unless otherwise specified.

Target power output 5 dBm using Dynamic RF Power Correction firmware routine per Section 14

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		RF Transmission Supply Current, T_A = -40°C					
		V_{DD} = 2.1V, PWR4:0 = 10000					
86	C	Data 1, FSK or ASK	I_{DD}	—	TBD	TBD	mA
87	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	4.45	4.75	mA
		V_{DD} = 2.5V, PWR4:0 = 10000					
88	C	Data 1, FSK or ASK	I_{DD}	—	TBD	TBD	mA
89	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	4.90	5.20	mA
		V_{DD} = 3.0V, PWR4:0 = 01111					
90	P	Data 1, FSK or ASK	I_{DD}	—	9.66	10.50	mA
91	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	5.46	5.75	mA
		V_{DD} = 3.3V, PWR4:0 = 01111					
92	C	Data 1, FSK or ASK	I_{DD}	—	10.00	10.85	mA
93	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	5.80	6.10	mA
		RF Transmission Supply Current, T_A = 25°C					
		V_{DD} = 2.1V, PWR4:0 = 01111					
94	C	Data 1, FSK or ASK	I_{DD}	—	9.50	10.25	mA
95	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	4.48	4.75	mA
		V_{DD} = 2.5V, PWR4:0 = 01111					
96	C	Data 1, FSK or ASK	I_{DD}	—	10.00	10.75	mA
97	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	4.92	5.20	mA
		V_{DD} = 3.0V, PWR4:0 = 01110					
98	P	Data 1, FSK or ASK	I_{DD}	—	10.28	11.00	mA
99	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	5.47	5.75	mA
		V_{DD} = 3.3V, PWR4:0 = 01110					
100	C	Data 1, FSK or ASK	I_{DD}	—	10.63	11.35	mA
101	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	5.80	6.10	mA
		RF Transmission Supply Current, T_A = 125°C					
		V_{DD} = 2.1V, PWR4:0 = 01111					
102	C	Data 1, FSK or ASK	I_{DD}	—	10.52	11.20	mA
103	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	4.40	4.65	mA
		V_{DD} = 2.5V, PWR4:0 = 01111					
104	C	Data 1, FSK or ASK	I_{DD}	—	11.00	11.75	mA
105	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	4.83	5.10	mA
		V_{DD} = 3.0V, PWR4:0 = 01110					
106	P	Data 1, FSK or ASK	I_{DD}	—	11.25	12.00	mA
107	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	5.36	5.66	mA
		V_{DD} = 3.3V, PWR4:0 = 01110					
108	C	Data 1, FSK or ASK	I_{DD}	—	11.62	12.40	mA
109	C	Data 0, ASK, Xtal oscillator, VCO, PLL Only	I_{DD}	—	5.67	6.00	mA
		RF Transmission Supply Current					
110	C	Delta I_{DD} per PWR4:0 bit step	ΔI_{DD}	—	0.5	—	mA

17.9 Control Timing

2.1 ≤ V_{DD} ≤ 3.6, T_L ≤ T_A ≤ T_H, unless otherwise specified.

Test load shown in Figure 17-1.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Internal Clock Frequency					
111	P	Initial startup frequency	f_{OSC}	6.00	8.00	10.00	MHz
112	P	Final frequency	f_{OSC}	7.44	8.00	8.56	MHz
113	C	PLL stabilization time to final frequency	t_{OSCSU}	—	300	1000	μSec
114	P	MCU Bus Frequency	f_{BUS}	—	0.5 f_{OSC}	—	MHz
115	P	LFR clock period (MFO)	$1/f_{MFO}$	7.44	8.00	8.56	μSec
		Low frequency clock period (LFO)					
116	C	Limited temperature range, -20 °C to +85 °C	$1/f_{LFO}$	800	1000	1200	μSec
117	C	Full temperature range, -40 °C to +125 °C	$1/f_{LFO}$	700	1000	1300	μSec
118	C	Charge pump oscillator (V_{DD} = 1.8 to 3.0 V)	f_{CPO}	1.4	2.0	2.6	MHz
		Power-On Reset Response					
119	C	Supply voltage rise time	t_{VDDR}	—	—	1	sec
120	C	Recovery time below V_{DD} = 0.5 V	t_{VDDOFF}	10	—	—	msec
		FLASH Programming Time					
121	C	8K using BDM interface, f_{OSC} = 8 MHz	t_{PROG}	—	0.5	1.0	Sec

17.10 Voltage Measurement Characteristics

2.3 ≤ V_{DD} ≤ 3.3, T_L ≤ T_A ≤ T_H, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Lower LVD detect threshold [1]					
122	P	V_{DD} falling	V_{LVDL}	1.82	1.88	1.93	V
123	P	V_{DD} rising	V_{LVDL}	1.92	1.96	2.01	V
124	C	Voltage drop detection time	t_{LVDL}	100	—	—	nsec
		Higher LVD detect threshold [1]					
125	P	V_{DD} falling	V_{LVDL}	2.07	2.13	2.18	V
126	P	V_{DD} rising	V_{LVDL}	2.16	2.21	2.26	V
127	C	Voltage drop detection time	t_{LVDL}	100	—	—	nsec
		Lower LVW detect threshold, LVWV = 0 [1]					
128	P	V_{DD} falling	V_{LVDL}	2.08	2.11	2.15	V
129	P	V_{DD} rising	V_{LVDL}	2.16	2.19	2.22	V
130	C	Voltage drop detection time	t_{LVDL}	100	—	—	nsec
		Higher LVW detect threshold, LVWV = 1 [1]					
131	P	V_{DD} falling	V_{LVDL}	2.35	2.40	2.45	V
132	P	V_{DD} rising	V_{LVDL}	2.35	2.40	2.45	V
133	C	Voltage drop detection time	t_{LVDL}	100	—	—	nsec
		Internal Voltage (V_{DD}, monotonic response) [2]					
134	C	REIMS_READ_COMP_VOLTAGE = 0	V	—	FAULT	—	V
135	C	REIMS_READ_COMP_VOLTAGE = 88	V	2.0	2.1	2.2	V
136	F	REIMS_READ_COMP_VOLTAGE = 128	V	2.4	2.5	2.6	V
137	C	REIMS_READ_COMP_VOLTAGE = 238	V	3.5	3.6	3.7	V
138	C	REIMS_READ_COMP_VOLTAGE = 255	V	—	FAULT	—	V
139	C	Voltage sensitivity at 25 °C, 2.5 V	ΔV	—	10	—	mV/count

NOTES
1. Low voltage detection and warning thresholds defined for voltage change rates less than 20 mV/μs.
2. Fully compensated measurements resulting from use of the included VCOMP firmware routine.

17.11 Temperature Measurement Characteristics

$2.3 \le V_{DD} \le 3.3$, $T_L \le T_A \le T_H$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Temperature measurement (monotonic response) [1] [2]					
140	C	REIMS_READ_COMP_TEMP_8 = 0	T	—	FAULT	—	°C
141	C	REIMS_READ_COMP_TEMP_8 = 15	T	-45	-40	-35	°C
142	C	REIMS_READ_COMP_TEMP_8 = 35	T	-23	-20	-17	°C
143	F	REIMS_READ_COMP_TEMP_8 = 80	T	22	25	28	°C
144	C	REIMS_READ_COMP_TEMP_8 = 125	T	67	70	73	°C
145	C	REIMS_READ_COMP_TEMP_8 = 180	T	120	125	130	°C
146	C	REIMS_READ_COMP_TEMP_8 = 255	T	—	FAULT	—	°C
147	C	Temperature sensitivity	ΔT	—	1.0	—	°C/count
148	C	Temperature measurement stability [3]	T_{STAB}	—	—	2	count
		Normal temperature restart					
149	C	Lower reset level	T_{RESET}	85	—	—	°C
150	C	Nominal threshold	T_{NORM}	—	105	—	°C
151	C	Higher rearm level	T_{REARM}	—	—	125	°C

NOTES:
1. Fully compensated measurements resulting from use of included TCOMP firmware routine.
2. Temperature error with MCU and RFX powered up at less than 10% duty-cycle.
3. Total range of variation over 30 consecutive measurements using compensated output format.
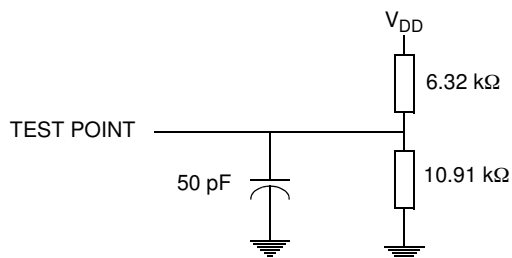

Figure 17-1 Control Timing Test Load for Digital Pins

17.12 Pressure Measurement Characteristic (100 to 450 kPa Range)

$2.3 \leq V_{DD} \leq 3.3$, $T_L \leq T_A \leq T_H$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Pressure Measurement [1]					
		0℃ ≤ T_A ≤ +70℃					
152	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
153	F	REIMS_READ_COMP_PRESSURE = 1	P	93	100	107	kPa
154	F	REIMS_READ_COMP_PRESSURE = 365	P	343	350	357	kPa
155	C	REIMS_READ_COMP_PRESSURE = 510	P	443	450	457	kPa
156	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
157	C	Press Sensitivity (100 - 450 kPa)	ΔP	—	0.686	—	kPa/cnt
		-20℃ ≤ T_A ≤ 0℃, 70℃ ≤ T_A ≤ 85℃					
158	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
159	F	REIMS_READ_COMP_PRESSURE = 1	P	89.5	100	110.5	kPa
160	F	REIMS_READ_COMP_PRESSURE = 365	P	339.5	350	360.5	kPa
161	C	REIMS_READ_COMP_PRESSURE = 510	P	439.5	450	460.5	kPa
162	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
163	C	Press Sensitivity (100 - 450 kPa)	ΔP	—	0.686	—	kPa/cnt
		-40℃ ≤ T_A ≤ -20℃, 85℃ ≤ T_A ≤ 125℃					
164	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
165	F	REIMS_READ_COMP_PRESSURE = 1	P	83.2	100	116.8	kPa
166	F	REIMS_READ_COMP_PRESSURE = 365	P	333.2	350	366.8	kPa
167	C	REIMS_READ_COMP_PRESSURE = 510	P	433.2	450	466.8	kPa
168	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
169	C	Press Sensitivity (100 - 450 kPa)	ΔP	—	0.686	—	kPa/cnt
170	C	Pressure measurement stability [2]	P_{STAB}	—	—	4	kPa

NOTES:
  1. Fully compensated measurements resulting from use of included PCOMP firmware routine.
  2. Total range of variation over 30 consecutive measurements using compensated output format.

17.13 Pressure Measurement Characteristic (100 to 800 kPa Range)

$2.3 \le V_{DD} \le 3.3$, $T_L \le T_A \le T_H$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Pressure Measurement [1]					
		$0\,°C \le T_A \le +70\,°C$					
171	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
172	F	REIMS_READ_COMP_PRESSURE = 1	P	90	100	110	kPa
173	F	REIMS_READ_COMP_PRESSURE = 401	P	640	650	660	kPa
174	C	REIMS_READ_COMP_PRESSURE = 510	P	790	800	810	kPa
175	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
176	C	Press Sensitivity (100 - 800 kPa)	ΔP	—	1.375	—	kPa/cnt
		$-20\,°C \le T_A \le 0\,°C$, $70\,°C \le T_A \le 85\,°C$					
177	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
178	F	REIMS_READ_COMP_PRESSURE = 1	P	85	100	115	kPa
179	F	REIMS_READ_COMP_PRESSURE = 401	P	635	650	665	kPa
180	C	REIMS_READ_COMP_PRESSURE = 510	P	785	800	815	kPa
181	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
182	C	Press Sensitivity (100 - 800 kPa)	ΔP	—	1.375	—	kPa/cnt
		$-40\,°C \le T_A \le -20\,°C$, $85\,°C \le T_A \le 125\,°C$					
183	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
184	F	REIMS_READ_COMP_PRESSURE = 1	P	76	100	124	kPa
185	F	REIMS_READ_COMP_PRESSURE = 401	P	626	650	674	kPa
186	C	REIMS_READ_COMP_PRESSURE = 510	P	776	800	824	kPa
187	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
188	C	Press Sensitivity (100 - 800 kPa)	ΔP	—	1.375	—	kPa/cnt
189	C	Pressure measurement stability [2]	P_{STAB}	—	—	5.5	kPa

NOTES:
1. Fully compensated measurements resulting from use of included PCOMP firmware routine.
2. Total range of variation over 30 consecutive measurements using compensated output format.

MPXY8300 Series

17.14 Pressure Measurement Characteristic (100 to 1500 kPa Range)

$2.3 \leq V_{DD} \leq 3.3$, $T_L \leq T_A \leq T_H$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Pressure Measurement [1]					
		$0\,°C \leq T_A \leq +70\,°C$					
190	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
191	F	REIMS_READ_COMP_PRESSURE = 1	P	80	100	120	kPa
192	F	REIMS_READ_COMP_PRESSURE = 137	P	480	500	520	kPa
193	C	REIMS_READ_COMP_PRESSURE = 341	P	1080	1100	1120	kPa
194	C	REIMS_READ_COMP_PRESSURE = 510	P	1480	1500	1520	kPa
195	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
196	C	Press Sensitivity (100 - 1500 kPa)	ΔP	—	2.941	—	kPa/cnt
		$-20\,°C \leq T_A \leq 0\,°C$, $70\,°C \leq T_A \leq 85\,°C$					
197	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
198	F	REIMS_READ_COMP_PRESSURE = 1	P	70	100	130	kPa
199	F	REIMS_READ_COMP_PRESSURE = 137	P	470	500	530	kPa
200	C	REIMS_READ_COMP_PRESSURE = 341	P	1070	1100	1130	kPa
201	C	REIMS_READ_COMP_PRESSURE = 510	P	1470	1500	1530	kPa
202	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
203	C	Press Sensitivity (100 - 1500 kPa)	ΔP	—	2.941	—	kPa/cnt
		$-40\,°C \leq T_A \leq -20\,°C$, $85\,°C \leq T_A \leq 125\,°C$					
204	C	REIMS_READ_COMP_PRESSURE = 0	P	—	FAULT	—	kPa
205	F	REIMS_READ_COMP_PRESSURE = 1	P	52	100	148	kPa
206	F	REIMS_READ_COMP_PRESSURE = 137	P	452	500	548	kPa
207	C	REIMS_READ_COMP_PRESSURE = 341	P	1052	1100	1148	kPa
208	C	REIMS_READ_COMP_PRESSURE = 510	P	1452	1500	1548	kPa
209	C	REIMS_READ_COMP_PRESSURE = 511	P	—	FAULT	—	kPa
210	C	Press Sensitivity (100 - 1500 kPa)	ΔP	—	2.941	—	kPa/cnt
211	C	Pressure measurement stability [2]	P_{STAB}	—	—	11.8	kPa

NOTES:
1. Fully compensated measurements resulting from use of included PCOMP firmware routine.
2. Total range of variation over 30 consecutive measurements using compensated output format

17.15 Acceleration Sensor Characteristics (X-Axis)

$2.3 \leq V_{DD} \leq 3.3$, $T_L \leq T_A \leq 90$ °C, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Acceleration Measurement [1]					
212	C	REIMS_READ_COMP_ACCEL_X = 0	A_X	—	FAULT	—	g
213	F	REIMS_READ_COMP_ACCEL_X = 1	A_X	-14	-10	-6	g
214	F	REIMS_READ_COMP_ACCEL_X = 255	A_X	-2	0	+2	g
215	F	REIMS_READ_COMP_ACCEL_X = 510	A_X	+6	+10	+14	g
216	C	REIMS_READ_COMP_ACCEL_X = 511	A_X	—	FAULT	—	g
217	C	Accel Sensitivity (±10 g)	ΔA_X	—	0.039	—	g/count
218	C	Acceleration measurement stability [2]	A_{STABX}	—	—	0.157	g
219	C	Acceleration cross-axis sensitivity	A_{CROSS}	-5	—	+5	%

17.16 Acceleration Sensor Characteristics (Z-Axis)

$2.3 \leq V_{DD} \leq 3.3$, $T_L \leq T_A \leq T_H$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Acceleration Measurement [1]					
220	C	REIMS_READ_COMP_ACCEL_Z = 0	A_Z	—	FAULT	—	g
221	F	REIMS_READ_COMP_ACCEL_Z = 1	A_Z	-5	0	+5	g
222	F	REIMS_READ_COMP_ACCEL_Z = 255	A_Z	+22	+30	+38	g
223	F	REIMS_READ_COMP_ACCEL_Z = 510	A_Z	+46	+60	+74	g
224	C	REIMS_READ_COMP_ACCEL_Z = 511	A_Z	—	FAULT	—	g
225	C	Accel Sensitivity (0 to 60 g)	ΔA_Z	—	0.118	—	g/count
226	C	Acceleration measurement stability [2]	A_{STABZ}	—	—	0.5	g
227	C	Acceleration cross-axis sensitivity	A_{CROSS}	-5	—	+5	%

NOTES:
1. Fully compensated measurements resulting from use of included ACOMP firmware routine with the 500 Hz low-pass filter.
2. Total range of variation over 30 consecutive fully compensated measurements resulting from use of included ACOMP firmware routine with the 500 Hz low-pass filter.

MPXY8300 Series

17.17 LFR Characteristics

2.1 ≤ V_{DD} ≤ 3.6, T_L ≤ T_A ≤ T_H, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		LFR Input Sensitivity, Low Sensitivity					
228	P	Detect level, PTA0:1	S_{DET}	—	—	10	mV p-p
229	P	No Detect level, PTA0:1	S_{NODET}	4	—	—	mV p-p
		LFR Input Detect Sensitivity, High Sensitivity					
230	P	Detect level, PTA0:1	S_{DET}	—	—	3.5	mV p-p
231	P	No Detect level, PTA0:1	S_{NODET}	0.5	—	—	mV p-p
		AGC Control (Manchester Data Mode, (Figure 17-2)					
232	C	Acquisition time for initial signal input	t_{ACQ}	—	—	1.25	μSec
233	C	Gain adjustment step	S_{AGC}	—	±14	—	%
234	C	Tracking delay for one gain adjustment step	t_{DELAY}	—	—	t_{DATA}	-
		LFR Modulation Depth (Manchester Data Mode)					
235	C	(Data 1 - Data 0)/Data 1	M_R	70	—	100	%
236	C	Maximum LFR Input (PTA0:1, Differential)	V_{LFIN}	-0.3	—	2.3	V
		LFR Input (Figure 17-3)					
237	C	Common Mode Impedance (R_1, R_2)	R_{LFCM}	0.5	—	—	MΩ
238	C	Differential Impedance	R_{LFR}	1	—	—	MΩ
239	C	Common-Mode Capacitance (C_1, C_2)	C_{LFR}	—	—	10	pF
240	C	Differential Capacitance (C_3)	C_{DLFR}	—	—	10	pF
		LFR Input Frequency					
241	C	Lower cutoff frequency of discriminator	f_{CBPL}	65	—	100	kHz
242	C	Upper cutoff frequency of discriminator	f_{CBPH}	150	—	250	kHz
		LFR Detector Sampling					
		Carrier Detect Mode On Time					
243	C	LFON = 0 (25 cycles of MFO)	t_{LFON}	—	200	—	μSec
244	C	LFON = 1 (265 cycles of MFO)	t_{LFON}	—	2120	—	μSec
		Manchester Data Mode On Time					
245	C	No signal detected	t_{LFON}	—	200	—	μSec
246	C	Signal detected, not acquired (2 MFO cycles)	t_{LFON}	—	2	—	mSec
247	C	Wakeup in middle of data frame	t_{LFON}	—	2	—	mSec
248	D	Series of Manchester zeros (168 data cycles)	$t_{TIMEOUT}$	45	—	—	mSec
		LFR Carrier Detect Time					
		LFCT = 0 (11 cycles of MFO)					
249	C	Detect	t_{CD_DET}	94.2	—	—	μSec
250	C	No Detect	t_{CD_NODET}	—	—	81.8	μSec
		LFCT = 1 (22 cycles of MFO)					
251	C	Detect	t_{CD_DET}	188.4	—	—	μSec
252	C	No Detect	t_{CD_NODET}	—	—	163.6	μSec
		LFR Decoder Timing (Manchester Encoded Mode)					
253	C	Guaranteed Data rate (Always decoded)	Dr	3.8	4.0	4.2	kbits/sec
254	C	Rejected Data rate (Never decoded)	Dr	<2.0		>6.5	kbits/sec
255	C	Data bit time	t_{DATA}	238	250	263	μSec
256	C	Pulse width	t_{PW}	—	0.50 t_{DATA}	—	-
257	C	Decoder extend time (cycles of MFO)	t_{LFDEC}	—	284	—	cycles
		Preamble time					
258	C	Time before synchronization	t_{LFPRE}	2 t_{DATA}	—	8 t_{DATA}	-
259	C	Maximum before LFR shutoff	t_{LFPRE}	—	—	180 t_{DATA}	-
260	C	LFR Mode Changeover Delay	t_{LFMODE}	—	—	4.29	mSec

MPXY8300 Series
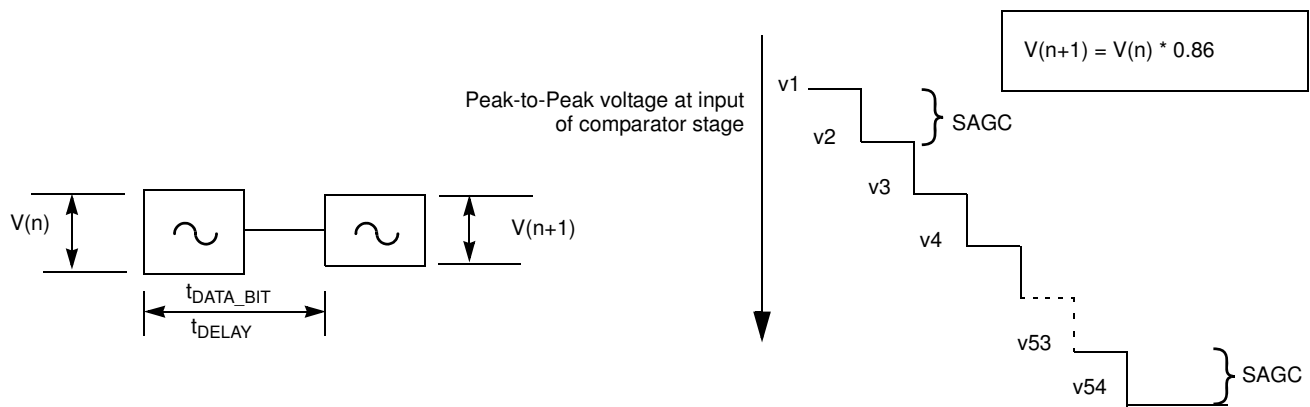
$$V(n+1) = V(n) * 0.86$$

Figure 17-2 AGC Tracking Delay and Adjustment Step
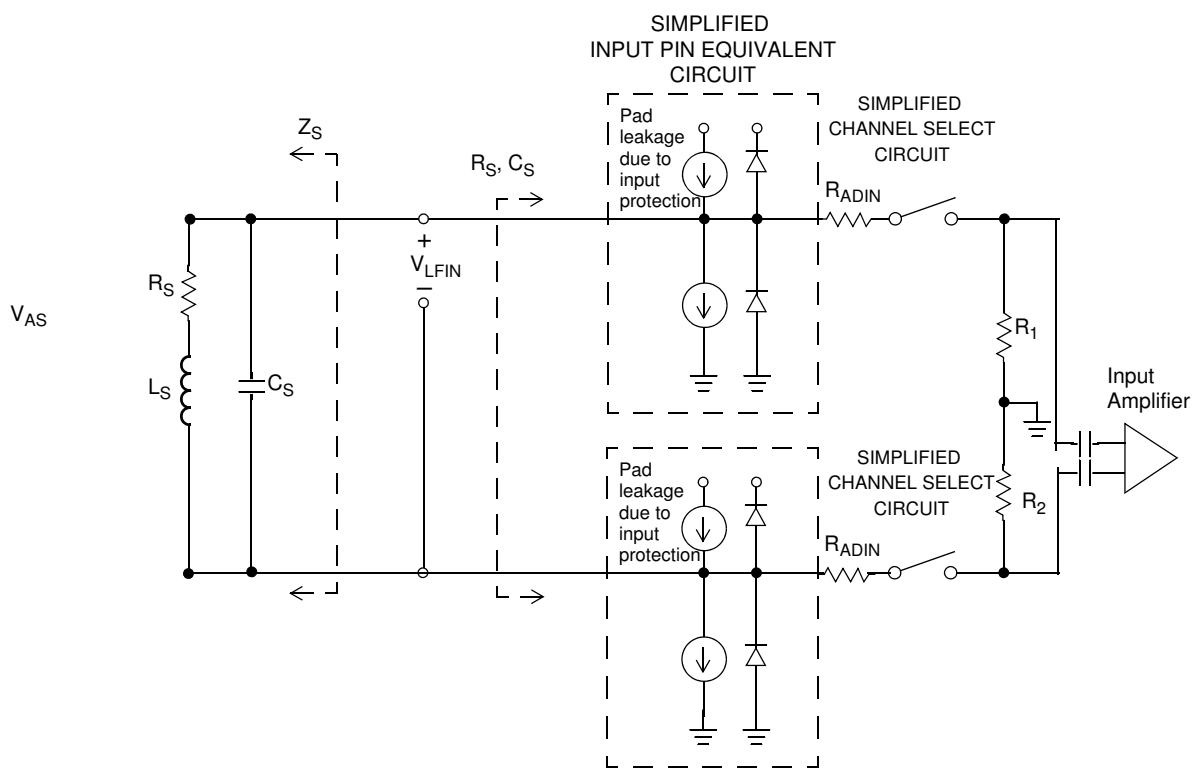

Figure 17-3 LFR Detector Input Equivalent Circuit

17.18 RF Output Stage

$2.1 \le V_{DD} \le 3.6$, $T_L \le T_A \le T_H$, unless otherwise specified.

Power output based on using Dynamic RF Power Correction firmware routine (Section 14)

Output load of 50 Ω resistance as shown in Figure 17-4 unless otherwise specified.

MCU in Stop1 mode during all RF tests.

V_{DD} powered setup in Figure 17-4.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
		Nominal Output Power with matching network					
		T_A = -40 °C					
261	C	V_{DD} = 2.1 V	P_{RF}	1.3	4.3	7.3	dBm
262	C	V_{DD} = 2.8 to 3.6 V	P_{RF}	2.0	5.0	8.0	dBm
		T_A = 25 °C					
263	C	V_{DD} = 2.1 V	P_{RF}	2.1	4.1	6.1	dBm
264	C	V_{DD} = 2.8 to 3.6 V	P_{RF}	3.0	5.0	7.0	dBm
		T_A = 125 °C					
265	C	V_{DD} = 2.1 V	P_{RF}	1.7	3.7	5.7	dBm
266	C	V_{DD} = 2.8 to 3.6 V	P_{RF}	3.0	5.0	7.0	dBm
		Programmable Power Adjustment Step PWR4:0 = 00000 through 11111					
267	C	-1 to 5 dBm	P_{ADJ}	—	0.5	—	dBm
		Programmable Frequency Steps Carrier and FSK Deviation (PLLCR3:0)					
268	C	CF = 0, 315 MHz nominal	f_{STEP}	—	0.97725	—	kHz
269	C	CF = 1, 434 MHZ nominal	f_{STEP}	—	1.34635	—	kHz
270	C	Harmonic 2 Level (315 and 434 MHz bands) with matching reference network	H2	—	-25	-19	dBc
271	C	Harmonic 3 Level (315 and 434 MHz bands) with matching reference network	H3	—	-40	-30	dBc
272	C	Spurious Noise (315 and 434 MHz bands) $f_{RF} \pm f_{REF}$	N_{SPUR}	—	-55	—	dBc
		Phase Noise (315 and 434 MHz bands) (T_A = 25 °C)					
273	C	$f_{RF} \pm 35$ kHz	N_{PH}	—	-85	-75	dBc/Hz
274	C	$f_{RF} \pm 390$ kHz	N_{PH}	—	-77	-73.5	dBc/Hz
275	C	Crystal Startup Time	t_{XTAL}	—	300	—	μsec
276	C	PLL Lock Time	t_{LOCK}	—	45	—	μsec
277	C	ASK Modulation Depth	M_{ASK}	60	—	—	dBc
278	C	XTAL Input Capacitance	C_{XTAL}	—	1	—	pF
279	C	XTAL Resistance	R_{XTAL}	—	—	150	ohm

Figure 17-4 Output Power Measurement Configuration

17.19 RFX Charge Pump

$2.1 \leq V_{DD} \leq 3.0$, $T_L \leq T_A \leq 25\,°C$, unless otherwise specified.

#	V	Characteristic	Symbol	Min	Typ	Max	Units
280	P	Charge Pump Oscillator frequency	f_{CPO}	—	10.0	—	MHz
281	P	Maximum charge voltage	V_{CAPMAX}	3.4	3.6	3.8	V
282	P	Maximum charge time (cycles of f_{CPO}) 2.1V to V_{CAPMAX}	t_{CHG}	—	—	100	msec
283	C	Average charge current from battery Charging V_{CAP} from 0 to V_{CAPMAX}	I_{BATT}	—	—	5	mA

Figure 17-5 Charge Pump Performance Test Configuration

SECTION 18   MECHANICAL SPECIFICATIONS

18.1   Parameter Identification and Validation

Each parameter in this section is indicated by a numerical number given in the "#" column.

The electrical parameters shown in this section are validated by various methods as designated in the "V" column. To give the customer a better understanding, the classifications given in Table 17-1 are used and the parameters are tagged accordingly.

18.2   Maximum Ratings

Maximum ratings are the extreme limits to which the device can be exposed without permanently damaging it. The device contains circuitry to protect the inputs against damage from high static voltages; however, do not apply voltages higher than those shown in the table below. Keep V_{IN} and V_{OUT} within the range $V_{SS} \leq (V_{IN}$ or $V_{OUT}) \leq V_{DD}$.

		Rating	Symbol	Value	Unit
		Maximum Pressure (absolute)			
284	C	Continuous, 450 kPa range	p_{max}	1400	kPa
285	C	Continuous, 800 kPa range	p_{max}	1800	kPa
286	C	Continuous, 1500 kPa range	p_{max}	2000	kPa
287	C	Pulsed, 5 seconds, 25°C, 450 & 800 kPa range	p_{max}	2500	kPa
288	C	Pulsed, 5 seconds, 25°C, 1500 kPa range	p_{max}	4000	kPa
		Centrifugal Force Effects			
289	C	Continuous acceleration (Z-axis)	g_{CENT}	2500	g
290	C	Powered Shock (peak, 0.1 msec, half-sine, 6 axis)	g_{shock}	6000	g
291	C	Drop Test (onto concrete, unpowered)	h_{DROP}	1.2	m
		Pressure Sensor Resonance			
292	D	Resonant frequency	f_{P0}	8	MHz
293	D	Damping Ratio	Q_P	1	
		X-axis Accelerometer Sensor Resonance			
294	D	Resonant frequency	f_{X0}	12.5	kHz
295	D	Damping Ratio	Q_X	0.1	
		Z-axis Accelerometer Sensor Resonance			
296	D	Resonant frequency (no-peak, overdamped)	f_{Z0}	9	kHz
297	D	Damping Ratio	Q_Z	5	
298	C	Package Weight	w	0.91	gm

18.3   Device Number

The current device numbers for the MPXY8300 Series series are given in Table 18-1.

Table 18-1 MPXY8300 Series Device Numbers

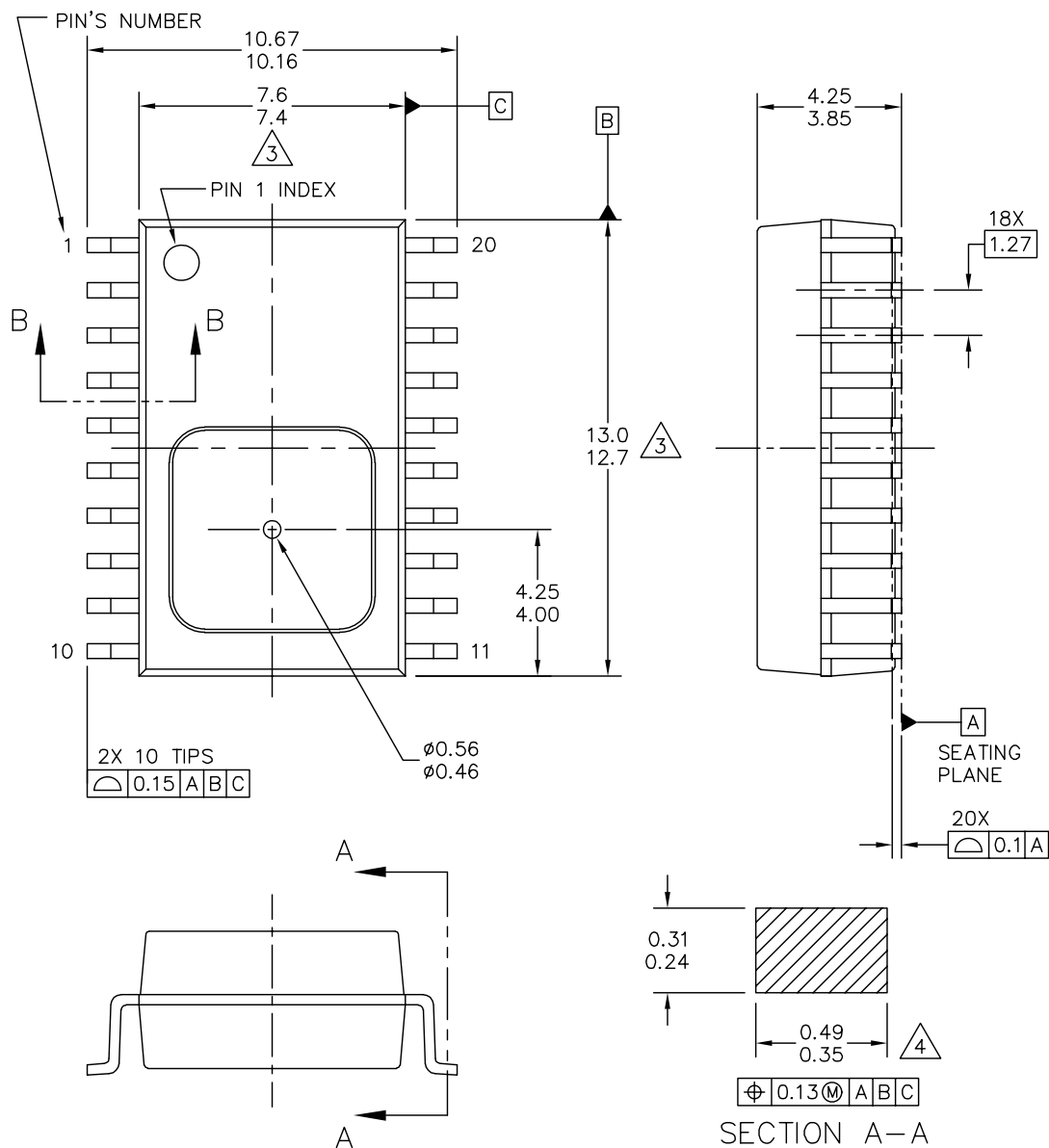| Sensing Capabilities					Pressure Range		
Press	Temp	Volt	Z-axis Accel	X-axis Accel	Low 100-450 kPa	Standard 100-800 kPa	High 100-1500 kPa
Yes	Yes	Yes	Yes	Yes	MPXY8310A	MPXY8300A	MPXY8320A
Yes	Yes	Yes	Yes	No	MPXY8310B	MPXY8300B	MPXY8320B
Yes	Yes	Yes	No	No	MPXY8310C	MPXY8300C	MPXY8320C

18.4 Media Compatibility

Media compatibility is as specified in Freescale SPD TPMS Media document, 12MSA1069D, Rev A.

18.5 Mounting Recommendations

The package should be mounted with the pressure port pointing away from the axis of tire rotation so that centripetal force will propel any contaminants out of the pressure port.

A plugged port will exhibit no change in pressure and can be cross checked in the user's software using the method described in Section 10.3.

MPXY8300 Series

Package Dimensions

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE	
TITLE: 20LD SOIC WIDE BODY PMCM CAVITY UP WITH FILTER CAP	DOCUMENT NO: 98ASA10701D	REV: B	
	CASE NUMBER: 1793-03	30 OCT 2007	
	STANDARD: NON-JEDEC		

PAGE 1 OF 3

CASE 1793-03
ISSUE B
20-LEAD SOIC

MPXY8300 Series

Sensors
Freescale Semiconductor

161

SECTION B—B

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE	
TITLE: 20LD SOIC WIDE BODY PMCM CAVITY UP WITH FILTER CAP	DOCUMENT NO: 98ASA10701D	REV: B	
	CASE NUMBER: 1793—03	30 OCT 2007	
	STANDARD: NON—JEDEC		

PAGE 2 OF 3

CASE 1793-03
ISSUE B
20-LEAD SOIC

MPXY8300 Series

NOTES:

1. ALL DIMENSIONS ARE IN MILLIMETERS.

2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M−1994.

⚠3. THIS DIMENSION DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS. MOLD FLASH OR PROTRUSIONS SHALL NOT EXCEED 0.15MM PER SIDE.

⚠4. THIS DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. PROTRUSIONS SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.75MM.

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE	
TITLE: 20LD SOIC WIDE BODY PMCM CAVITY UP WITH FILTER CAP	DOCUMENT NO: 98ASA10701D	REV: B	
	CASE NUMBER: 1793−03	30 OCT 2007	
	STANDARD: NON−JEDEC		

PAGE 3 OF 3

CASE 1793-03
ISSUE B
20-LEAD SOIC

MPXY8300 Series

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 010 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

freescale™
semiconductor