



Welcome to E-XFL.COM

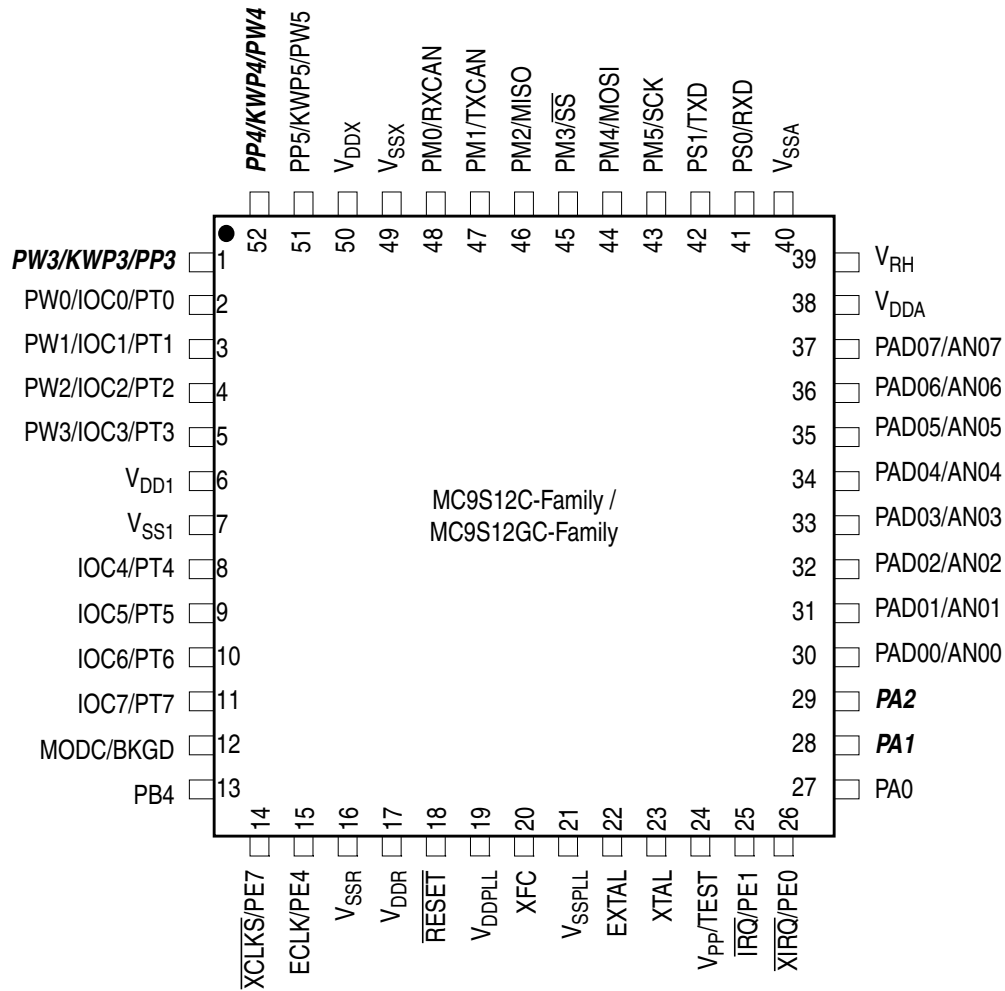
What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

| | |
|----------------------------|---|
| Product Status | Obsolete |
| Core Processor | HCS12 |
| Core Size | 16-Bit |
| Speed | 25MHz |
| Connectivity | EBI/EMI, SCI, SPI |
| Peripherals | POR, PWM, WDT |
| Number of I/O | 60 |
| Program Memory Size | 96KB (96K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 4K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.35V ~ 5.5V |
| Data Converters | A/D 8x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 125°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 80-QFP |
| Supplier Device Package | 80-QFP (14x14) |
| Purchase URL | https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc9s12gc96mfue |



* Signals shown in ***Bold italic*** are not available on the 48-pin package

Figure 1-8. Pin Assignments in 52-Pin LQFP



2.3.2.3.6 Port M Polarity Select Register (PPSM)

Module Base + 0x0015

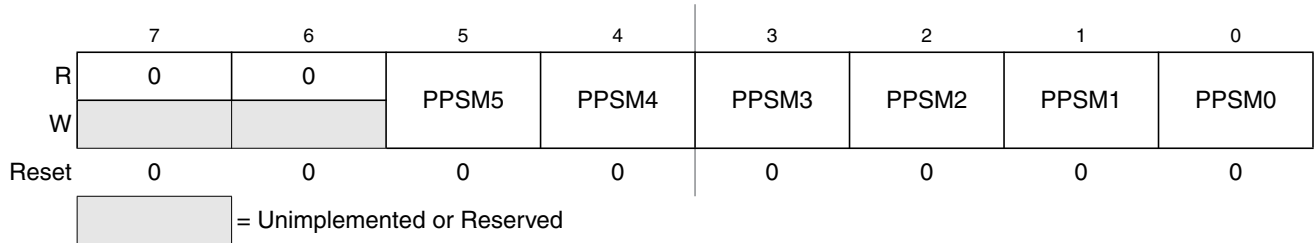


Figure 2-22. Port M Polarity Select Register (PPSM)

Read: Anytime.

Write: Anytime.

Table 2-20. PPSM Field Descriptions

| Field | Description |
|------------------|--|
| 5–0 PPSM[5:0] | Polarity Select Port M — This register selects whether a pull-down or a pull-up device is connected to the pin. 0 A pull-up device is connected to the associated port M pin, if enabled by the associated bit in register PERM and if the port is used as input or as wired-or output. 1 A pull-down device is connected to the associated port M pin, if enabled by the associated bit in register PERM and if the port is used as input. |

2.3.2.3.7 Port M Wired-OR Mode Register (WOMM)

Module Base + 0x0016

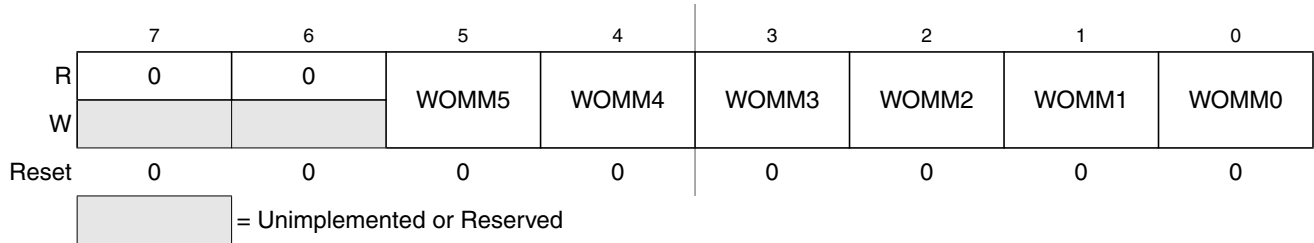


Figure 2-23. Port M Wired-OR Mode Register (WOMM)

Read: Anytime.

Write: Anytime.

Table 2-21. WOMM Field Descriptions

| Field | Description |
|------------------|--|
| 5–0 WOMM[5:0] | Wired-OR Mode Port M — This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven. This bit has no influence on pins used as inputs. 0 Output buffers operate as push-pull outputs. 1 Output buffers operate as open-drain outputs. |

firmware. The standard BDM firmware watches for serial commands and executes them as they are received.

The firmware commands are shown in [Table 6-6](#).

Table 6-6. Firmware Commands

| Command ⁽¹⁾ | Opcode (hex) | Data | Description |
|-------------------------|--------------|-----------------|--|
| READ_NEXT | 62 | 16-bit data out | Increment X by 2 ($X = X + 2$), then read word X points to. |
| READ_PC | 63 | 16-bit data out | Read program counter. |
| READ_D | 64 | 16-bit data out | Read D accumulator. |
| READ_X | 65 | 16-bit data out | Read X index register. |
| READ_Y | 66 | 16-bit data out | Read Y index register. |
| READ_SP | 67 | 16-bit data out | Read stack pointer. |
| WRITE_NEXT | 42 | 16-bit data in | Increment X by 2 ($X = X + 2$), then write word to location pointed to by X. |
| WRITE_PC | 43 | 16-bit data in | Write program counter. |
| WRITE_D | 44 | 16-bit data in | Write D accumulator. |
| WRITE_X | 45 | 16-bit data in | Write X index register. |
| WRITE_Y | 46 | 16-bit data in | Write Y index register. |
| WRITE_SP | 47 | 16-bit data in | Write stack pointer. |
| GO | 08 | None | Go to user program. If enabled, ACK will occur when leaving active background mode. |
| GO_UNTIL ⁽²⁾ | 0C | None | Go to user program. If enabled, ACK will occur upon returning to active background mode. |
| TRACE1 | 10 | None | Execute one user instruction then return to active BDM. If enabled, ACK will occur upon returning to active background mode. |
| TAGGO | 18 | None | Enable tagging and go to user program. There is no ACK pulse related to this command. |

1. If enabled, ACK will occur when data is ready for transmission for all BDM READ commands and will occur after the write is complete for all BDM WRITE commands.
2. Both WAIT (with clocks to the S12 CPU core disabled) and STOP disable the ACK function. The GO_UNTIL command will not get an Acknowledge if one of these two CPU instructions occurs before the "UNTIL" instruction. This can be a problem for any instruction that uses ACK, but GO_UNTIL is a lot more difficult for the development tool to time-out.

6.4.5 BDM Command Structure

Hardware and firmware BDM commands start with an 8-bit opcode followed by a 16-bit address and/or a 16-bit data word depending on the command. All the read commands return 16 bits of data despite the byte or word implication in the command name.

NOTE

8-bit reads return 16-bits of data, of which, only one byte will contain valid data. If reading an even address, the valid data will appear in the MSB. If reading an odd address, the valid data will appear in the LSB.

Figure 6-11 shows the ACK handshake protocol in a command level timing diagram. The READ_BYTE instruction is used as an example. First, the 8-bit instruction opcode is sent by the host, followed by the address of the memory location to be read. The target BDM decodes the instruction. A bus cycle is grabbed (free or stolen) by the BDM and it executes the READ_BYTE operation. Having retrieved the data, the BDM issues an ACK pulse to the host controller, indicating that the addressed byte is ready to be retrieved. After detecting the ACK pulse, the host initiates the byte retrieval process. Note that data is sent in the form of a word and the host needs to determine which is the appropriate byte based on whether the address was odd or even.

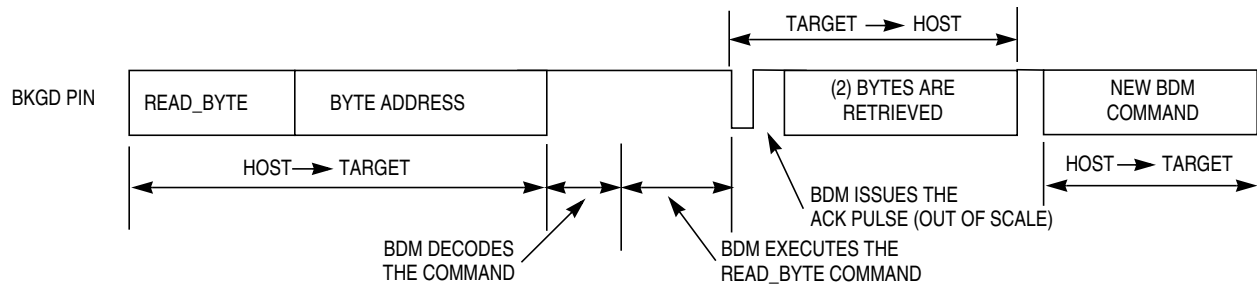


Figure 6-11. Handshake Protocol at Command Level

Differently from the normal bit transfer (where the host initiates the transmission), the serial interface ACK handshake pulse is initiated by the target MCU by issuing a falling edge in the BKGD pin. The hardware handshake protocol in Figure 6-10 specifies the timing when the BKGD pin is being driven, so the host should follow this timing constraint in order to avoid the risk of an electrical conflict in the BKGD pin.

NOTE

The only place the BKGD pin can have an electrical conflict is when one side is driving low and the other side is issuing a speedup pulse (high). Other “highs” are pulled rather than driven. However, at low rates the time of the speedup pulse can become lengthy and so the potential conflict time becomes longer as well.

The ACK handshake protocol does not support nested ACK pulses. If a BDM command is not acknowledge by an ACK pulse, the host needs to abort the pending command first in order to be able to issue a new BDM command. When the CPU enters WAIT or STOP while the host issues a command that requires CPU execution (e.g., WRITE_BYTE), the target discards the incoming command due to the WAIT or STOP being detected. Therefore, the command is not acknowledged by the target, which means that the ACK pulse will not be issued in this case. After a certain time the host should decide to abort the ACK sequence in order to be free to issue a new command. Therefore, the protocol should provide a mechanism in which a command, and therefore a pending ACK, could be aborted.

NOTE

Differently from a regular BDM command, the ACK pulse does not provide a time out. This means that in the case of a WAIT or STOP instruction being executed, the ACK would be prevented from being issued. If not aborted, the ACK would remain pending indefinitely. See the handshake abort procedure described in Section 6.4.8, “Hardware Handshake Abort Procedure.”

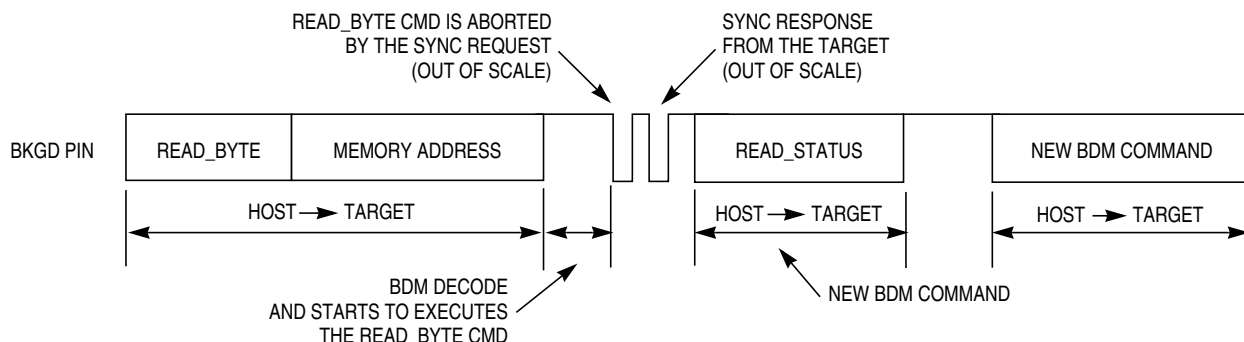


Figure 6-12. ACK Abort Procedure at the Command Level

Figure 6-13 shows a conflict between the ACK pulse and the SYNC request pulse. This conflict could occur if a POD device is connected to the target BKGD pin and the target is already in debug active mode. Consider that the target CPU is executing a pending BDM command at the exact moment the POD is being connected to the BKGD pin. In this case, an ACK pulse is issued along with the SYNC command. In this case, there is an electrical conflict between the ACK speedup pulse and the SYNC pulse. Because this is not a probable situation, the protocol does not prevent this conflict from happening.

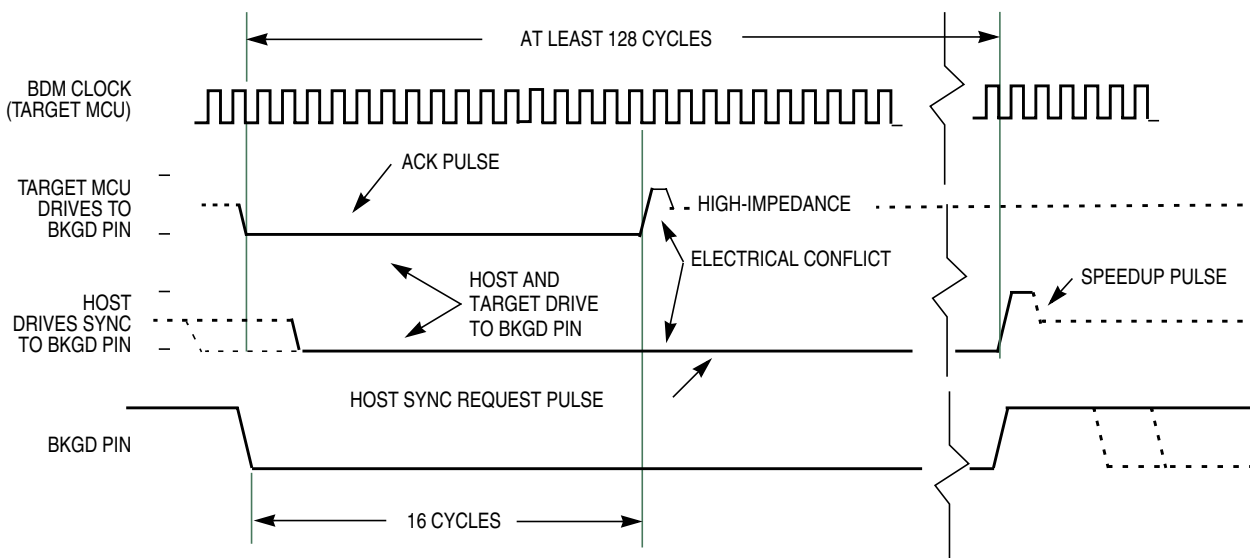


Figure 6-13. ACK Pulse and SYNC Request Conflict

NOTE

This information is being provided so that the MCU integrator will be aware that such a conflict could eventually occur.

The hardware handshake protocol is enabled by the `ACK_ENABLE` and disabled by the `ACK_DISABLE` BDM commands. This provides backwards compatibility with the existing POD devices which are not able to execute the hardware handshake protocol. It also allows for new POD devices, that support the hardware handshake protocol, to freely communicate with the target device. If desired, without the need for waiting for the ACK pulse.

If a read command is issued but the data is not retrieved within 512 serial clock cycles, a soft-reset will occur causing the command to be disregarded. The data is not available for retrieval after the time-out has occurred. This is the expected behavior if the handshake protocol is not enabled. However, consider the behavior where the BDC is running in a frequency much greater than the CPU frequency. In this case, the command could time out before the data is ready to be retrieved. In order to allow the data to be retrieved even with a large clock frequency mismatch (between BDC and CPU) when the hardware handshake protocol is enabled, the time out between a read command and the data retrieval is disabled. Therefore, the host could wait for more than 512 serial clock cycles and continue to be able to retrieve the data from an issued read command. However, as soon as the handshake pulse (ACK pulse) is issued, the time-out feature is re-activated, meaning that the target will time out after 512 clock cycles. Therefore, the host needs to retrieve the data within a 512 serial clock cycles time frame after the ACK pulse had been issued. After that period, the read command is discarded and the data is no longer available for retrieval. Any falling edge of the BKGD pin after the time-out period is considered to be a new command or a SYNC request.

Note that whenever a partially issued command, or partially retrieved data, has occurred the time out in the serial communication is active. This means that if a time frame higher than 512 serial clock cycles is observed between two consecutive negative edges and the command being issued or data being retrieved is not complete, a soft-reset will occur causing the partially received command or data retrieved to be disregarded. The next falling edge of the BKGD pin, after a soft-reset has occurred, is considered by the target as the start of a new BDM command, or the start of a SYNC request pulse.

6.4.13 Operation in Wait Mode

The BDM cannot be used in wait mode if the system disables the clocks to the BDM.

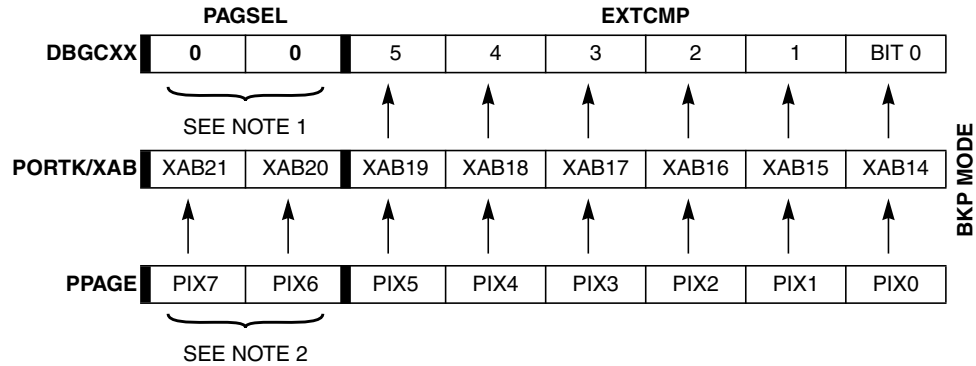
There is a clearing mechanism associated with the WAIT instruction when the clocks to the BDM (CPU core platform) are disabled. As the clocks restart from wait mode, the BDM receives a soft reset (clearing any command in progress) and the ACK function will be disabled. This is a change from previous BDM modules.

6.4.14 Operation in Stop Mode

The BDM is completely shutdown in stop mode.

There is a clearing mechanism associated with the STOP instruction. STOP must be enabled and the part must go into stop mode for this to occur. As the clocks restart from stop mode, the BDM receives a soft reset (clearing any command in progress) and the ACK function will be disabled. This is a change from previous BDM modules.





- NOTES:
1. In BKP mode, PAGSEL has no functionality. Therefore, set PAGSEL to 00 (reset state).
 2. Current HCS12 implementations are limited to six PPAGE bits, PIX[5:0].

Figure 7-16. Comparators A and B Extended Comparison in BKP Mode

7.3.2.10 Debug Comparator A Register (DBGCA)

Module Base + 0x002B
Starting address location affected by INITRG register setting.

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|-------|--------|--------|--------|--------|--------|--------|-------|-------|
| R | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Figure 7-17. Debug Comparator A Register High (DBGCAH)

Module Base + 0x002C
Starting address location affected by INITRG register setting.

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

Figure 7-18. Debug Comparator A Register Low (DBGCAL)

Table 7-21. DBGCA Field Descriptions

| Field | Description |
|-------|--|
| 15:0 | Comparator A Compare Bits — The comparator A compare bits control whether comparator A compares the address bus bits [15:0] to a logic 1 or logic 0. See Table 7-20 . |
| 15:0 | 0 Compare corresponding address bit to a logic 0 |
| | 1 Compare corresponding address bit to a logic 1 |

8.3.2.3 ATD Control Register 2 (ATDCTL2)

This register controls power down, interrupt, and external trigger. Writes to this register will abort current conversion sequence but will not start a new sequence.

Module Base + 0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-----------------------------|------|------|---------|--------|--------|-------|-------|
| R | ADPU | AFFC | AWAI | ETRIGLE | ETRIGP | ETRIGE | ASCIE | ASCIF |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | = Unimplemented or Reserved | | | | | | | |

Figure 8-5. ATD Control Register 2 (ATDCTL2)

Read: Anytime

Write: Anytime

Table 8-1. ATDCTL2 Field Descriptions

| Field | Description |
|--------------|--|
| 7 ADPU | ATD Power Down — This bit provides on/off control over the ATD10B8C block allowing reduced MCU power consumption. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after ADPU bit is enabled. 0 Power down ATD 1 Normal ATD functionality |
| 6 AFFC | ATD Fast Flag Clear All 0 ATD flag clearing operates normally (read the status register ATDSTAT1 before reading the result register to clear the associate CCF flag). 1 Changes all ATD conversion complete flags to a fast clear sequence. Any access to a result register will cause the associate CCF flag to clear automatically. |
| 5 AWAI | ATD Power Down in Wait Mode — When entering Wait Mode this bit provides on/off control over the ATD10B8C block allowing reduced MCU power. Because analog electronic is turned off when powered down, the ATD requires a recovery time period after exit from Wait mode. 0 ATD continues to run in Wait mode 1 Halt conversion and power down ATD during Wait mode After exiting Wait mode with an interrupt conversion will resume. But due to the recovery time the result of this conversion should be ignored. |
| 4 ETRIGLE | External Trigger Level/Edge Control — This bit controls the sensitivity of the external trigger signal. See Table 8-2 for details. |
| 3 ETRIGP | External Trigger Polarity — This bit controls the polarity of the external trigger signal. See Table 8-2 for details. |
| 2 ETRIGE | External Trigger Mode Enable — This bit enables the external trigger on ATD channel 7. The external trigger allows to synchronize sample and ATD conversions processes with external events. 0 Disable external trigger 1 Enable external trigger Note: The conversion results for the external trigger ATD channel 7 have no meaning while external trigger mode is enabled. |

9.3.2.2 CRG Reference Divider Register (REFDV)

The REFDV register provides a finer granularity for the PLL multiplier steps. The count in the reference divider divides OSCCLK frequency by REFDV + 1.

Module Base + 0x0001

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|--------|--------|--------|--------|
| R | 0 | 0 | 0 | 0 | REFDV3 | REFDV2 | REFDV1 | REFDV0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |


 = Unimplemented or Reserved

Figure 9-5. CRG Reference Divider Register (REFDV)

Read: anytime

Write: anytime except when PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit and the track detector bit.

9.3.2.3 Reserved Register (CTFLG)

This register is reserved for factory testing of the CRGV4 module and is not available in normal modes.

Module Base + 0x0002

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---|---|---|
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

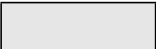
 = Unimplemented or Reserved

Figure 9-6. CRG Reserved Register (CTFLG)

Read: always reads 0x0000 in normal modes

Write: unimplemented in normal modes

NOTE

Writing to this register when in special mode can alter the CRGV4 functionality.

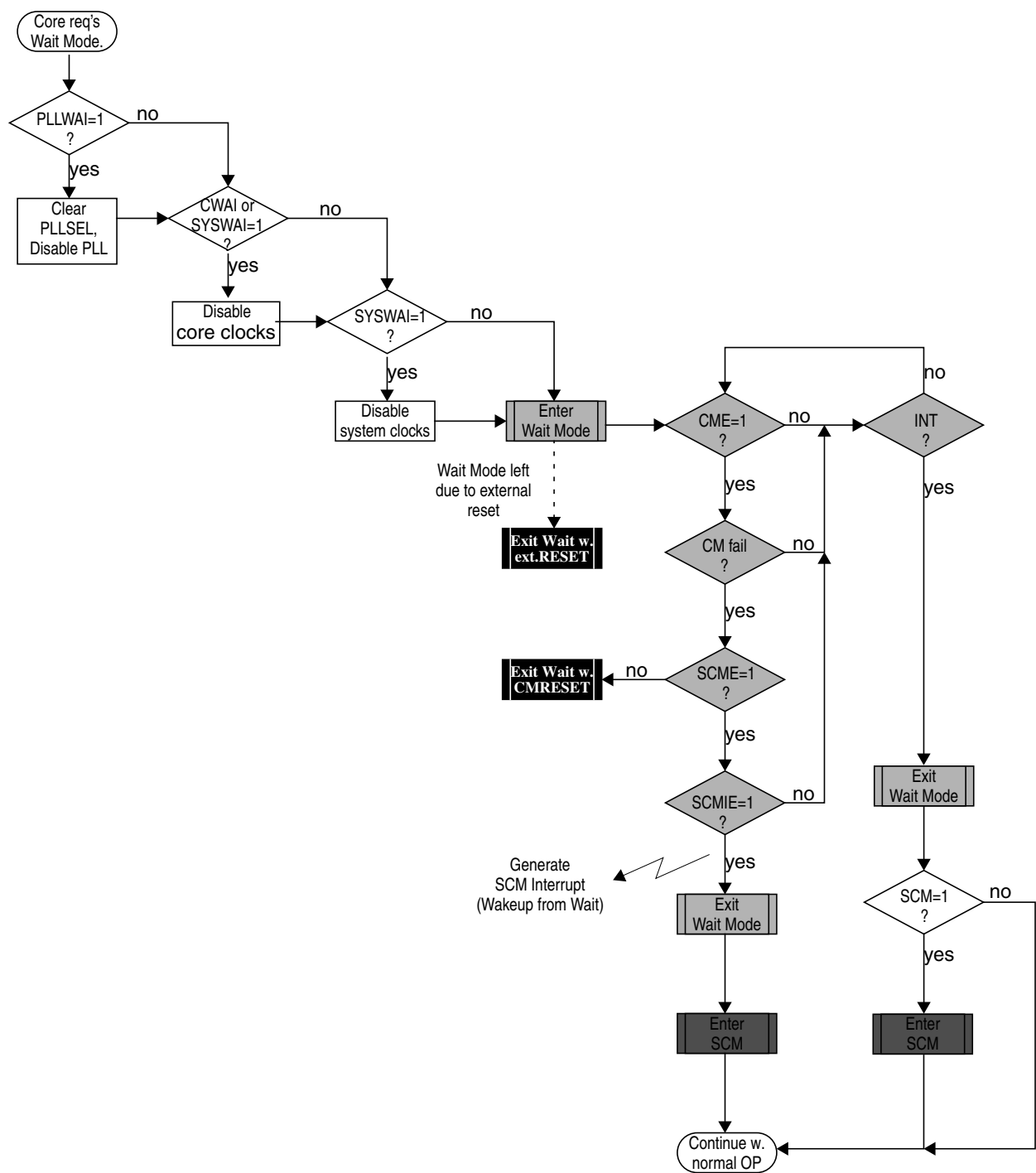


Figure 9-23. Wait Mode Entry/Exit Sequence

Table 13-11. Start Bit Verification

| RT3, RT5, and RT7 Samples | Start Bit Verification | Noise Flag |
|---------------------------|------------------------|------------|
| 100 | Yes | 1 |
| 101 | No | 0 |
| 110 | No | 0 |
| 111 | No | 0 |

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. [Table 13-12](#) summarizes the results of the data bit samples.

Table 13-12. Data Bit Recovery

| RT8, RT9, and RT10 Samples | Data Bit Determination | Noise Flag |
|----------------------------|------------------------|------------|
| 000 | 0 | 0 |
| 001 | 0 | 1 |
| 010 | 0 | 1 |
| 011 | 1 | 1 |
| 100 | 0 | 1 |
| 101 | 1 | 1 |
| 110 | 1 | 1 |
| 111 | 1 | 0 |

NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. [Table 13-13](#) summarizes the results of the stop bit samples.

Table 13-13. Stop Bit Recovery

| RT8, RT9, and RT10 Samples | Framing Error Flag | Noise Flag |
|----------------------------|--------------------|------------|
| 000 | 1 | 0 |
| 001 | 1 | 1 |
| 010 | 1 | 1 |
| 011 | 0 | 1 |
| 100 | 1 | 1 |
| 101 | 0 | 1 |
| 110 | 0 | 1 |
| 111 | 0 | 0 |

Table 14-2. SPICR1 Field Descriptions

| Field | Description |
|------------|---|
| 7 SPIE | SPI Interrupt Enable Bit — This bit enables SPI interrupt requests, if SPIF or MODF status flag is set. 0 SPI interrupts disabled. 1 SPI interrupts enabled. |
| 6 SPE | SPI System Enable Bit — This bit enables the SPI system and dedicates the SPI port pins to SPI system functions. If SPE is cleared, SPI is disabled and forced into idle state, status bits in SPISR register are reset. 0 SPI disabled (lower power consumption). 1 SPI enabled, port pins are dedicated to SPI functions. |
| 5 SPTIE | SPI Transmit Interrupt Enable — This bit enables SPI interrupt requests, if SPTEF flag is set. 0 SPTEF interrupt disabled. 1 SPTEF interrupt enabled. |
| 4 MSTR | SPI Master/Slave Mode Select Bit — This bit selects, if the SPI operates in master or slave mode. Switching the SPI from master to slave or vice versa forces the SPI system into idle state. 0 SPI is in slave mode 1 SPI is in master mode |
| 3 CPOL | SPI Clock Polarity Bit — This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Active-high clocks selected. In idle state SCK is low. 1 Active-low clocks selected. In idle state SCK is high. |
| 2 CPHA | SPI Clock Phase Bit — This bit is used to select the SPI clock format. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Sampling of data occurs at odd edges (1,3,5,...,15) of the SCK clock 1 Sampling of data occurs at even edges (2,4,6,...,16) of the SCK clock |
| 1 SSOE | Slave Select Output Enable — The \overline{SS} output feature is enabled only in master mode, if MODFEN is set, by asserting the SSOE as shown in Table 14-3. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. |
| 0 LSBFE | LSB-First Enable — This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Data is transferred most significant bit first. 1 Data is transferred least significant bit first. |

Table 14-3. \overline{SS} Input / Output Selection

| MODFEN | SSOE | Master Mode | Slave Mode |
|--------|------|---|-----------------------|
| 0 | 0 | \overline{SS} not used by SPI | \overline{SS} input |
| 0 | 1 | \overline{SS} not used by SPI | \overline{SS} input |
| 1 | 0 | \overline{SS} input with MODF feature | \overline{SS} input |
| 1 | 1 | \overline{SS} is slave select output | \overline{SS} input |

Table 18-9. Flash Protection Function

| FPOPEN | FPHDIS | FPHS[1] | FPHS[0] | FPLDIS | FPLS[1] | FPLS[0] | Function ⁽¹⁾ |
|--------|--------|---------|---------|--------|---------|---------|---------------------------------|
| 1 | 1 | x | x | 1 | x | x | No protection |
| 1 | 1 | x | x | 0 | x | x | Protect low range |
| 1 | 0 | x | x | 1 | x | x | Protect high range |
| 1 | 0 | x | x | 0 | x | x | Protect high and low ranges |
| 0 | 1 | x | x | 1 | x | x | Full Flash array protected |
| 0 | 0 | x | x | 1 | x | x | Unprotected high range |
| 0 | 1 | x | x | 0 | x | x | Unprotected low range |
| 0 | 0 | x | x | 0 | x | x | Unprotected high and low ranges |

1. For range sizes refer to [Table 18-10](#) and [Table 18-11](#).

Table 18-10. Flash Protection Higher Address Range

| FPHS[1:0] | Address Range | Range Size |
|-----------|---------------|------------|
| 00 | 0xF800–0xFFFF | 2 Kbytes |
| 01 | 0xF000–0xFFFF | 4 Kbytes |
| 10 | 0xE000–0xFFFF | 8 Kbytes |
| 11 | 0xC000–0xFFFF | 16 Kbytes |

Table 18-11. Flash Protection Lower Address Range

| FPLS[1:0] | Address Range | Range Size |
|-----------|---------------|------------|
| 00 | 0x4000–0x41FF | 512 bytes |
| 01 | 0x4000–0x43FF | 1 Kbyte |
| 10 | 0x4000–0x47FF | 2 Kbytes |
| 11 | 0x4000–0x4FFF | 4 Kbytes |

Figure 18-9 illustrates all possible protection scenarios. Although the protection scheme is loaded from the Flash array after reset, it is allowed to change in normal modes. This protection scheme can be used by applications requiring re-programming in single chip mode while providing as much protection as possible if no re-programming is required.

Table 18-14. FCMD Field Descriptions

| Field | Description |
|---|---|
| 6, 5, 2, 0 CMDB[6:5] CMDB[2] CMDB[0] | Valid Flash commands are shown in Table 18-15 . An attempt to execute any command other than those listed in Table 18-15 will set the ACCERR bit in the FSTAT register (see Section 18.3.2.6). |

Table 18-15. Valid Flash Command List

| CMDB | NVM Command |
|------|--------------|
| 0x05 | Erase verify |
| 0x20 | Word program |
| 0x40 | Sector erase |
| 0x41 | Mass erase |

18.3.2.8 RESERVED2

This register is reserved for factory testing and is not accessible to the user.

Module Base + 0x0007

| | | | | | | | | |
|-------|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

Figure 18-12. RESERVED2

All bits read 0 and are not writable.

18.3.2.9 Flash Address Register (FADDR)

FADDRHI and FADDRLO are the Flash address registers.

Module Base + 0x0008

| | | | | | | | | |
|-------|---|---|-------|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | 0 | 0 | FABHI | | | | | |
| W | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

= Unimplemented or Reserved

Figure 18-13. Flash Address High Register (FADDRHI)

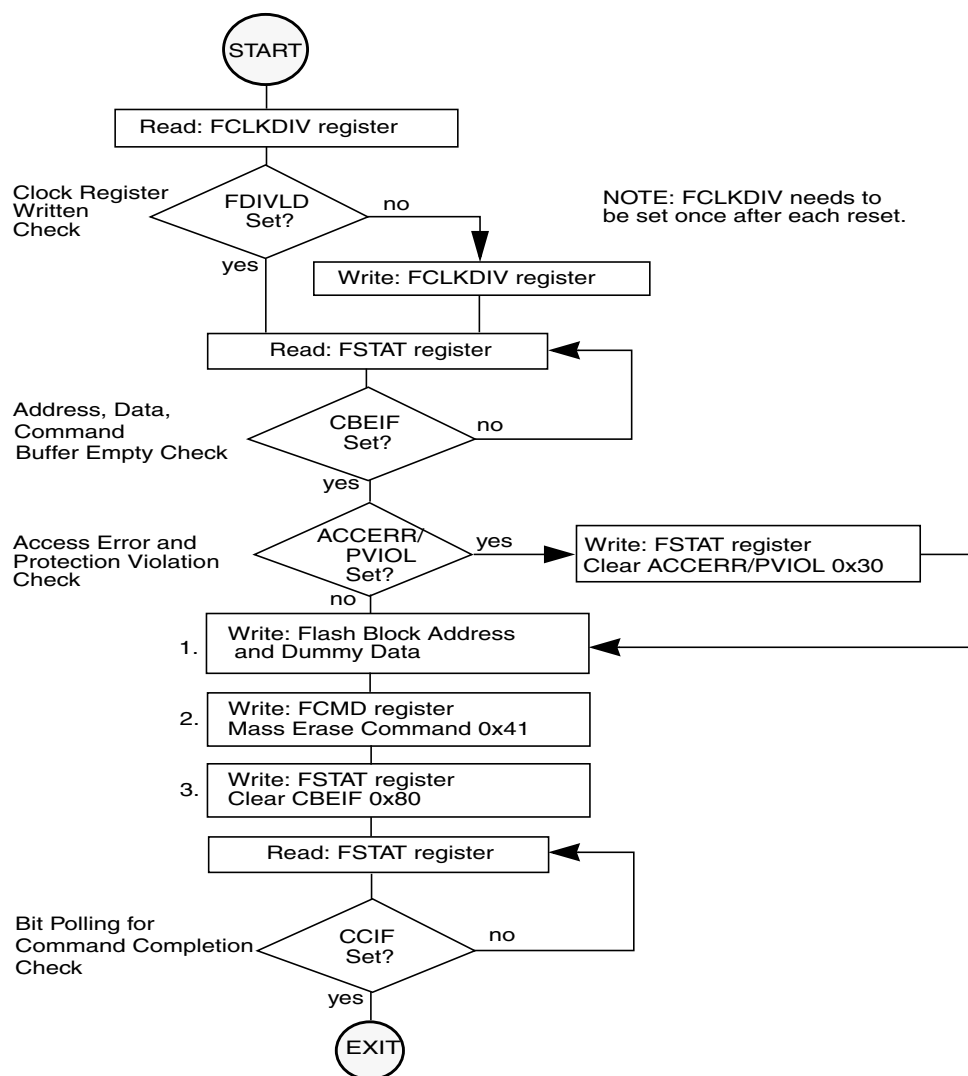
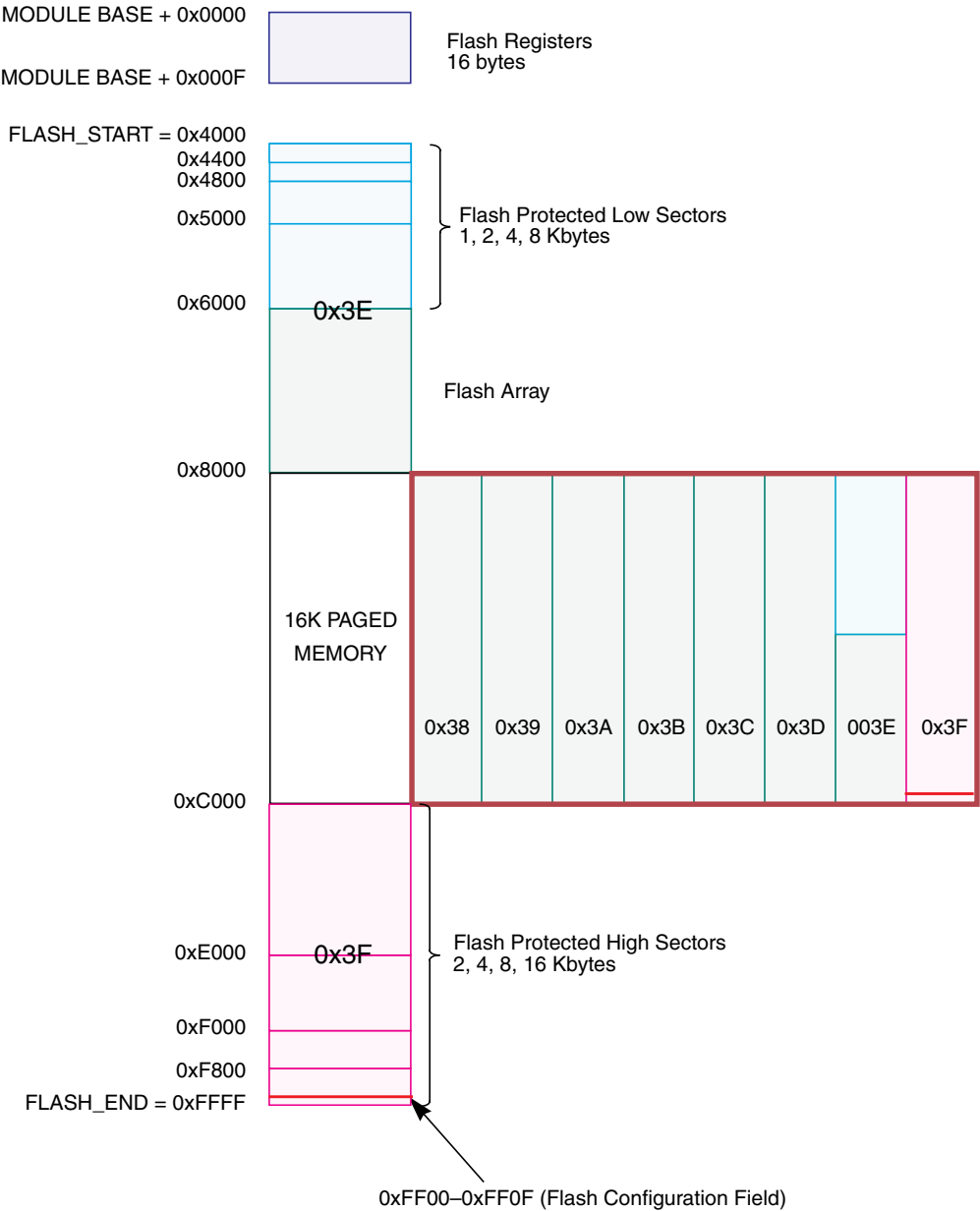


Figure 18-25. Example Mass Erase Command Flow



Note: 0x38–0x3F correspond to the PPAGE register content

Figure 19-3. Flash Memory Map

Chapter 20

96 Kbyte Flash Module (S12FTS96KV1)

20.1 Introduction

The **FTS128K1FTS96K** module implements a **12896** Kbyte Flash (nonvolatile) memory. The Flash memory contains one array of **12896** Kbytes organized as **1024768** rows of **128128** bytes with an erase sector size of eight rows (**10241024** bytes). The Flash array may be read as either bytes, aligned words, or misaligned words. Read access time is one bus cycle for byte and aligned word, and two bus cycles for misaligned words.

The Flash array is ideal for program and data storage for single-supply applications allowing for field reprogramming without requiring external voltage sources for program or erase. Program and erase functions are controlled by a command driven interface. The Flash module supports both mass erase and sector erase. An erased bit reads 1 and a programmed bit reads 0. The high voltage required to program and erase is generated internally. It is not possible to read from a Flash array while it is being erased or programmed.

CAUTION

A Flash word must be in the erased state before being programmed.
Cumulative programming of bits within a Flash word is not allowed.

20.1.1 Glossary

Command Write Sequence — A three-step MCU instruction sequence to program, erase, or erase verify the Flash array memory.

20.1.2 Features

- **12896** Kbytes of Flash memory comprised of one **12896** Kbyte array divided into **12896** sectors of **10241024** bytes
- Automated program and erase algorithm
- Interrupts on Flash command completion and command buffer empty
- Fast sector erase and word program operation
- 2-stage command pipeline for faster multi-word program times
- Flexible protection scheme to prevent accidental program or erase
- Single power supply for Flash program and erase operations
- Security feature to prevent unauthorized access to the Flash array memory

A.4.3 Phase Locked Loop

The oscillator provides the reference clock for the PLL. The PLL's Voltage Controlled Oscillator (VCO) is also the system clock source in self clock mode.

A.4.3.1 XFC Component Selection

This section describes the selection of the XFC components to achieve a good filter characteristics.

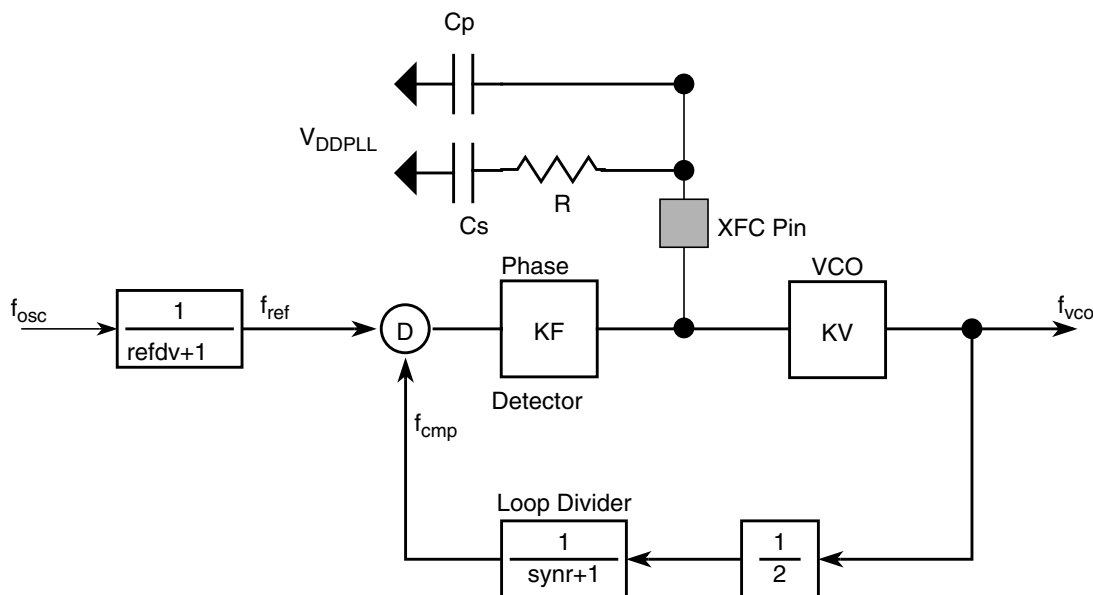


Figure A-2. Basic PLL Functional Diagram

The following procedure can be used to calculate the resistance and capacitance values using typical values for K_1 , f_1 and i_{ch} from Table A-17.

The grey boxes show the calculation for $f_{VCO} = 50\text{MHz}$ and $f_{ref} = 1\text{MHz}$. E.g., these frequencies are used for $f_{OSC} = 4\text{MHz}$ and a 25MHz bus clock.

The VCO Gain at the desired VCO frequency is approximated by:

$$K_V = K_1 \cdot e^{\frac{(f_1 - f_{VCO})}{K_1 \cdot 1V}} = -100 \cdot e^{\frac{(60 - 50)}{-100}} = -90.48\text{MHz/V}$$

The phase detector relationship is given by:

$$K_\Phi = -|i_{ch}| \cdot K_V = 316.7\text{Hz}/\Omega$$

i_{ch} is the current in tracking mode.