Section 4. Architecture

HIGHLIGHTS

This section of the manual contains the following major topics:

4.1 Introduction

The high performance of the PICmicro™ devices can be attributed to a number of architectural features commonly found in RISC microprocessors. These include:

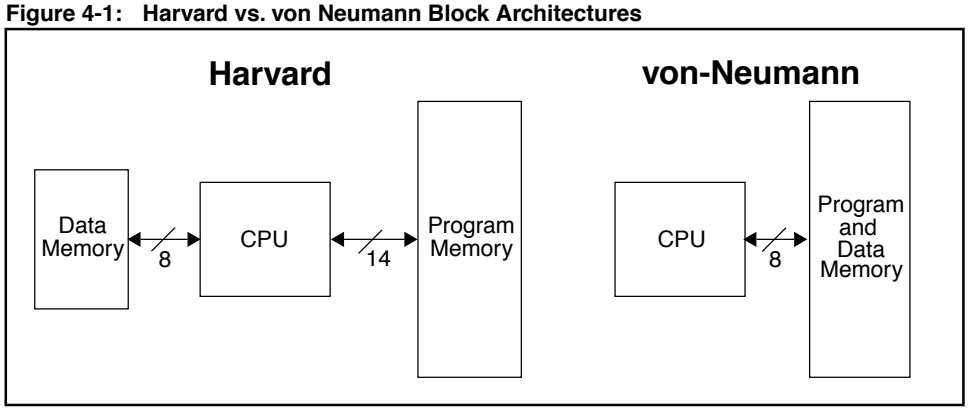
- Harvard architecture
- Long Word Instructions
- Single Word Instructions
- Single Cycle Instructions
- Instruction Pipelining
- Reduced Instruction Set
- Register File Architecture
- Orthogonal (Symmetric) Instructions

Figure 4-2 shows a simple core memory bus arrangement for Mid-Range MCU devices.

Harvard Architecture:

Harvard architecture has the program memory and data memory as separate memories and are accessed from separate buses. This improves bandwidth over traditional von Neumann architecture in which program and data are fetched from the same memory using the same bus. To execute an instruction, a von Neumann machine must make one or more (generally more) accesses across the 8-bit bus to fetch the instruction. Then data may need to be fetched, operated on, and possibly written. As can be seen from this description, that bus can be extremely conjested. While with a Harvard architecture, the instruction is fetched in a single instruction cycle (all 14-bits). While the program memory is being accessed, the data memory is on an independent bus and can be read and written. These separated buses allow one instruction to execute while the next instruction is fetched. A comparison of Harvard vs. von-Neumann architectures is shown in Figure 4-1.

Figure 4-1: Harvard vs. von Neumann Block Architectures



Long Word Instructions:

Long word instructions have a wider (more bits) instruction bus than the 8-bit Data Memory Bus. This is possible because the two buses are separate. This further allows instructions to be sized differently than the 8-bit wide data word which allows a more efficient use of the program memory, since the program memory width is optimized to the architectural requirements.

Single Word Instructions:

Single Word instruction opcodes are 14-bits wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single cycle. With single word instructions, the number of words of program memory locations equals the number of instructions for the device. This means that all locations are valid instructions.

Typically in the von Neumann architecture, most instructions are multi-byte. In general, a device with 4-KBytes of program memory would allow approximately 2K of instructions. This 2:1 ratio is generalized and dependent on the application code. Since each instruction may take multiple bytes, there is no assurance that each location is a valid instruction.

Instruction Pipeline:

The instruction pipeline is a two-stage pipeline which overlaps the fetch and execution of instructions. The fetch of the instruction takes one TCY, while the execution takes another TCY. However, due to the overlap of the fetch of current instruction and execution of previous instruction, an instruction is fetched and another instruction is executed every single TCY.

Single Cycle Instructions:

With the Program Memory bus being 14-bits wide, the entire instruction is fetched in a single machine cycle (TCY). The instruction contains all the information required and is executed in a single cycle. There may be a one cycle delay in execution if the result of the instruction modified the contents of the Program Counter. This requires the pipeline to be flushed and a new instruction to be fetched.

Reduced Instruction Set:

When an instruction set is well designed and highly orthogonal (symmetric), fewer instructions are required to perform all needed tasks. With fewer instructions, the whole set can be more rapidly learned.

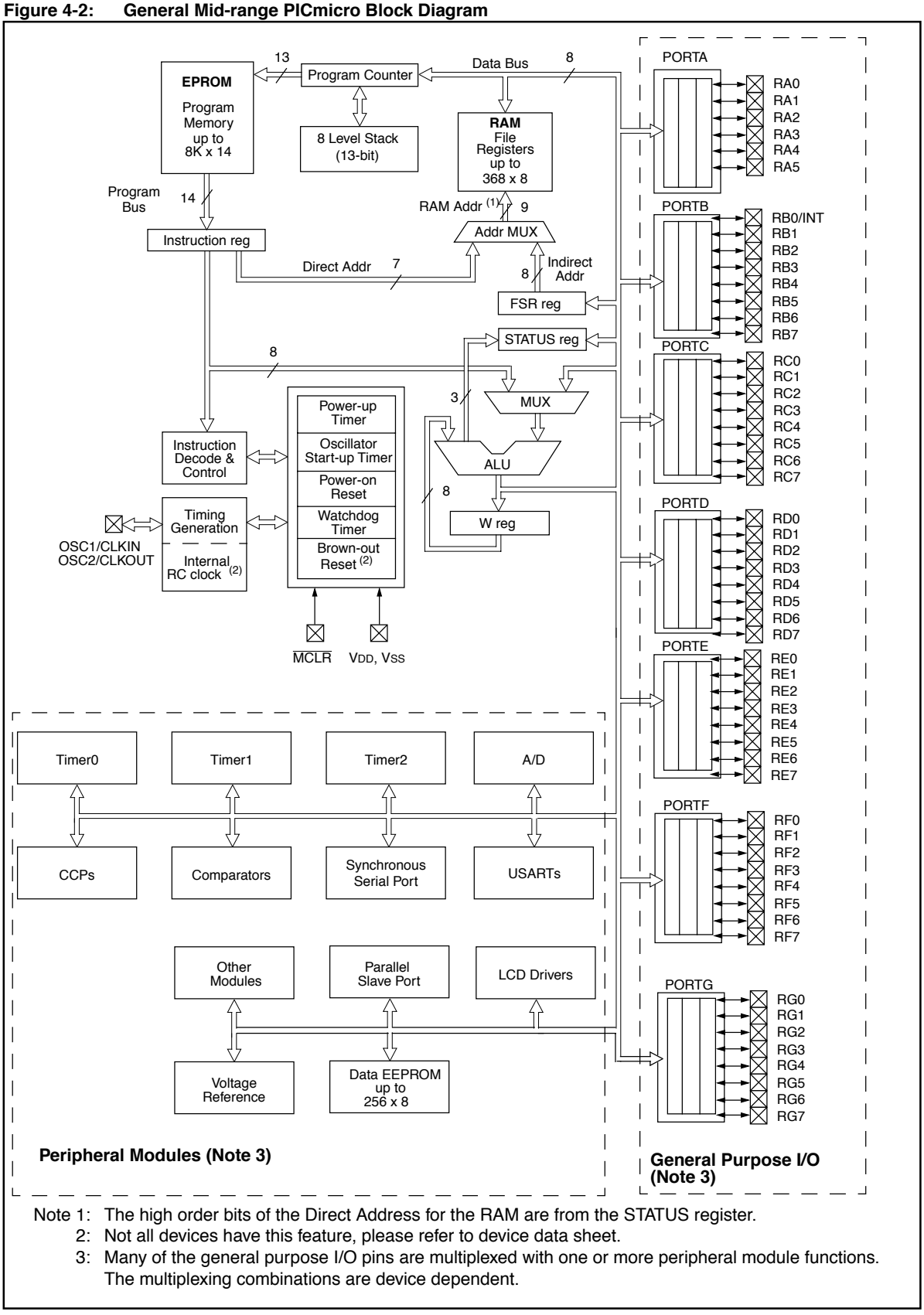
Register File Architecture:

The register files/data memory can be directly or indirectly addressed. All special function registers, including the program counter, are mapped in the data memory.

Orthogonal (Symmetric) Instructions:

Orthogonal instructions make it possible to carry out any operation on any register using any addressing mode. This symmetrical nature and lack of “special instructions” make programming simple yet efficient. In addition, the learning curve is reduced significantly. The mid-range instruction set uses only two non-register oriented instructions, which are used for two of the cores features. One is the SLEEP instruction which places the device into the lowest power use mode. The other is the CLRWDT instruction which verifies the chip is operating properly by preventing the on-chip Watchdog Timer (WDT) from overflowing and resetting the device.

Figure 4-2: General Mid-range PICmicro Block Diagram

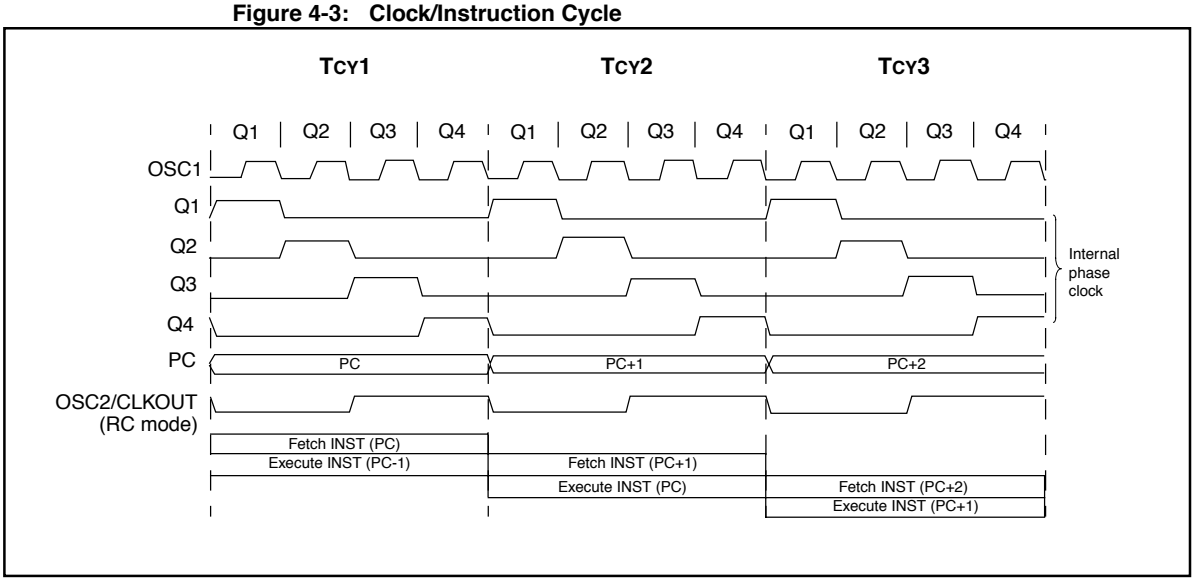


Note 1: The high order bits of the Direct Address for the RAM are from the STATUS register.
2: Not all devices have this feature, please refer to device data sheet.
3: Many of the general purpose I/O pins are multiplexed with one or more peripheral module functions. The multiplexing combinations are device dependent.

4.2 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3, and Q4. Internally, the program counter (PC) is incremented every Q1, and the instruction is fetched from the program memory and latched into the instruction register in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are illustrated in Figure 4-3, and Example 4-1.

Figure 4-3: Clock/Instruction Cycle



4.3 Instruction Flow/Pipelining

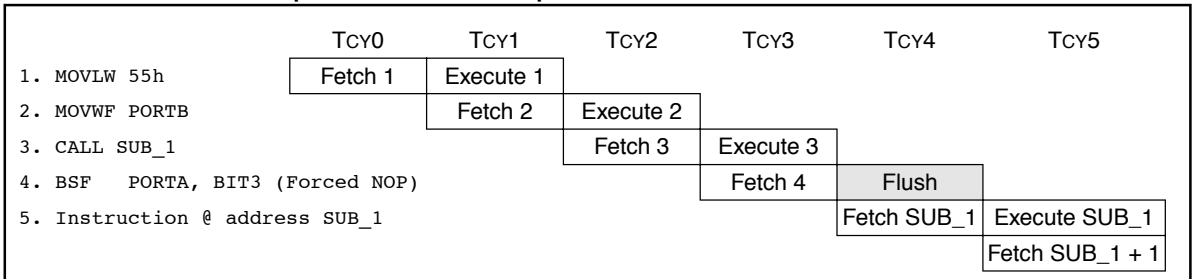
An "Instruction Cycle" consists of four Q cycles (Q1, Q2, Q3, and Q4). Fetch takes one instruction cycle while decode and execute takes another instruction cycle. However, due to Pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g. GOTO) then an extra cycle is required to complete the instruction (Example 4-1).

The instruction **fetch** begins with the program counter incrementing in Q1.

In the **execution** cycle, the fetched instruction is latched into the "Instruction Register (IR)" in cycle Q1. This instruction is then decoded and executed during the Q2, Q3, and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

Example 4-1 shows the operation of the two stage pipeline for the instruction sequence shown. At time TCY0, the first instruction is fetched from program memory. During TCY1, the first instruction executes while the second instruction is fetched. During TCY2, the second instruction executes while the third instruction is fetched. During TCY3, the fourth instruction is fetched while the third instruction (CALL SUB_1) is executed. When the third instruction completes execution, the CPU forces the address of instruction four onto the Stack and then changes the Program Counter (PC) to the address of SUB_1. This means that the instruction that was fetched during TCY3 needs to be "flushed" from the pipeline. During TCY4, instruction four is flushed (executed as a NOP) and the instruction at address SUB_1 is fetched. Finally during TCY5, instruction five is executed and the instruction at address SUB_1 + 1 is fetched.

Example 4-1: Instruction Pipeline Flow



All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is "flushed" from the pipeline while the new instruction is being fetched and then executed.

4.4 I/O Descriptions

Table 4-1 gives a brief description of the functions that may be multiplexed to a port pin. Multiple functions may exist on one port pin. When multiplexing occurs, the peripheral module's functional requirements may force an override of the data direction (TRIS bit) of the port pin (such as in the A/D and LCD modules).

Table 4-1: I/O Descriptions

Pin Name	Pin Type	Buffer Type	Description
			Analog Input Channels
AN0	I	Analog	
AN1	I	Analog	
AN2	I	Analog	
AN3	I	Analog	
AN4	I	Analog	
AN5	I	Analog	
AN6	I	Analog	
AN7	I	Analog	
AN8	I	Analog	
AN9	I	Analog	
AN10	I	Analog	
AN11	I	Analog	
AN12	I	Analog	
AN13	I	Analog	
AN14	I	Analog	
AN15	I	Analog	
AVDD	P	P	Analog Power
AVSS	P	P	Analog Ground
C1	I	Analog	LCD Voltage Generation
C2	I	Analog	LCD Voltage Generation
CCP1	I/O	ST	Capture1 input/Compare1 output/PWM1 output
CCP2	I/O	ST	Capture2 input/Compare2 output/PWM2 output.
CDAC	O	Analog	A/D ramp current source output. Normally connected to external capacitor to generate a linear voltage ramp.
CK	I/O	ST	USART Synchronous Clock, always associated with TX pin function (See related TX, RX, DT)
CLKIN	I	ST/CMOS	External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKIN, OSC2/CLKOUT pins)
CLKOUT	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate. Always associated with OSC2 pin function. (See related OSC2, OSC1)
CMPA	O	—	Comparator A output
CMPB	O	—	Comparator B output

Legend: TTL = TTL-compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
SM = SMBus compatible input. An external resistor is required if this pin is used as an output
NPU = N-channel pull-up
PU = Weak internal pull-up
No-P diode = No P-diode to VDD
AN = Analog input or output
I = input
O = output
P = Power
L = LCD Driver

Table 4-1: I/O Descriptions (Cont.'d)

Pin Name	Pin Type	Buffer Type	Description
COM0	L	—	LCD Common Driver0
COM1	L	—	LCD Common Driver1
COM2	L	—	LCD Common Driver2
COM3	L	—	LCD Common Driver3
$\overline{CS}$	I	TTL	chip select control for parallel slave port (See related $\overline{RD}$ and $\overline{WR}$)
DT	I/O	ST	USART Synchronous Data. Always associated RX pin function. (See related RX, TX, CK)
			GP is a bi-directional I/O port. Some pins of port GP can be software programmed for internal weak pull-ups on the inputs.
GP0	I/O	TTL/ST	TTL input buffer as general purpose I/O, Schmitt Trigger input buffer when used as the serial programming mode.
GP1	I/O	TTL/ST	TTL input buffer as general purpose I/O, Schmitt Trigger input buffer when used as the serial programming mode.
GP2	I/O	ST	
GP3	I	TTL	
GP4	I/O	TTL	
GP5	I/O	TTL	
INT	I	ST	External Interrupt
$\overline{MCLR}/V_{PP}$	I/P	ST	Master clear (reset) input or programming voltage input. This pin is an active low reset to the device.
NC	—	—	These pins should be left unconnected.
OSC1	I	ST/CMOS	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode. CMOS otherwise.
OSC2	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
PBTN	I	ST	Input with weak pull-up resistor, can be used to generate an interrupt.
PSP0	I/O	TTL	Parallel Slave Port for interfacing to a microprocessor port. These pins have TTL input buffers when PSP module is enabled.
PSP1	I/O	TTL	
PSP2	I/O	TTL	
PSP3	I/O	TTL	
PSP4	I/O	TTL	
PSP5	I/O	TTL	
PSP6	I/O	TTL	
PSP7	I/O	TTL	
			PORTA is a bi-directional I/O port.
RA0	I/O	TTL	
RA1	I/O	TTL	
RA2	I/O	TTL	
RA3	I/O	TTL	
RA4	I/O	ST	RA4 is an open drain when configured as output.
RA5	I/O	TTL	

Legend: TTL = TTL-compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
SM = SMBus compatible input. An external resistor is required if this pin is used as an output
NPU = N-channel pull-up
PU = Weak internal pull-up
No-P diode = No P-diode to V_{DD}
AN = Analog input or output
I = input
O = output
P = Power
L = LCD Driver

Table 4-1: I/O Descriptions (Cont.'d)

Pin Name	Pin Type	Buffer Type	Description
			PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0	I/O	TTL	
RB1	I/O	TTL	
RB2	I/O	TTL	
RB3	I/O	TTL	
RB4	I/O	TTL	Interrupt on change pin.
RB5	I/O	TTL	Interrupt on change pin.
RB6	I/O	TTL/ST	Interrupt on change pin. Serial programming clock. TTL input buffer as general purpose I/O, Schmitt Trigger input buffer when used as the serial programming clock.
RB7	I/O	TTL/ST	Interrupt on change pin. Serial programming data. TTL input buffer as general purpose I/O, Schmitt Trigger input buffer when used as the serial programming data.
			PORTC is a bi-directional I/O port.
RC0	I/O	ST	
RC1	I/O	ST	
RC2	I/O	ST	
RC3	I/O	ST	
RC4	I/O	ST	
RC5	I/O	ST	
RC6	I/O	ST	
RC7	I/O	ST	
$\overline{RD}$	I	TTL	Read control for parallel slave port (See also $\overline{WR}$ and $\overline{CS}$ pins)
			PORTD is a bi-directional I/O port.
RD0	I/O	ST	
RD1	I/O	ST	
RD2	I/O	ST	
RD3	I/O	ST	
RD4	I/O	ST	
RD5	I/O	ST	
RD6	I/O	ST	
RD7	I/O	ST	
			PORTE is a bi-directional I/O port.
RE0	I/O	ST	
RE1	I/O	ST	
RE2	I/O	ST	
RE3	I/O	ST	
RE4	I/O	ST	
RE5	I/O	ST	
RE6	I/O	ST	
RE7	I/O	ST	

Legend: TTL = TTL-compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
SM = SMBus compatible input. An external resistor is required if this pin is used as an output
NPU = N-channel pull-up
PU = Weak internal pull-up
No-P diode = No P-diode to V_{DD}
AN = Analog input or output
I = input
O = output
P = Power
L = LCD Driver

Table 4-1: I/O Descriptions (Cont.'d)

Pin Name	Pin Type	Buffer Type	Description
REFA	O	CMOS	Programmable reference A output.
REFB	O	CMOS	Programmable reference B output.
			PORTF is a digital input or LCD Segment Driver Port
RF0	I/O	ST	
RF1	I/O	ST	
RF2	I/O	ST	
RF3	I/O	ST	
RF4	I/O	ST	
RF5	I/O	ST	
RF6	I/O	ST	
RF7	I/O	ST	
			PORTG is a digital input or LCD Segment Driver Port
RG0	I/O	ST	
RG1	I/O	ST	
RG2	I/O	ST	
RG3	I/O	ST	
RG4	I/O	ST	
RG5	I/O	ST	
RG6	I/O	ST	
RG7	I/O	ST	
RX	I	ST	USART Asynchronous Receive
SCL	I/O	ST	Synchronous serial clock input/output for I^2C mode.
SCLA	I/O	ST	Synchronous serial clock for I^2C interface.
SCLB	I/O	ST	Synchronous serial clock for I^2C interface.
SDA	I/O	ST	I^2C™ Data I/O
SDAA	I/O	ST	Synchronous serial data I/O for I^2C interface
SDAB	I/O	ST	Synchronous serial data I/O for I^2C interface
SCK	I/O	ST	Synchronous serial clock input/output for SPI mode.
SDI	I	ST	SPI Data In
SDO	O	—	SPI Data Out (SPI mode)
$\overline{SS}$	I	ST	SPI Slave Select input
SEG00 to SEG31	I/L	ST	LCD Segment Driver00 through Driver31.
SUM	O	AN	AN1 summing junction output. This pin can be connected to an external capacitor for averaging small duration pulses.
T0CKI	I	ST	Timer0 external clock input
T1CKI	I	ST	Timer1 external clock input
T1OSO	O	CMOS	Timer1 oscillator output
T1OSI	I	CMOS	Timer1 oscillator input
TX	O	—	USART Asynchronous Transmit (See related RX)

Legend: TTL = TTL-compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
SM = SMBus compatible input. An external resistor is required if this pin is used as an output
NPU = N-channel pull-up
PU = Weak internal pull-up
No-P diode = No P-diode to V_{DD}
AN = Analog input or output
I = input
O = output
P = Power
L = LCD Driver

I^2C is a trademark of Philips Corporation.

Table 4-1: I/O Descriptions (Cont.'d)

Pin Name	Pin Type	Buffer Type	Description
VLCD1	P	—	LCD Voltage
VLCD2	P	—	LCD Voltage
VLCD3	P	—	LCD Voltage
VLCDADJ	I	Analog	LCD Voltage Generation
VREF	I	Analog	Analog High Voltage Reference input. DR reference voltage output on devices with comparators.
VREF+	I	Analog	Analog High Voltage Reference input. Usually multiplexed onto an analog pin.
VREF-	I	Analog	Analog Low Voltage Reference input. Usually multiplexed onto an analog pin.
VREG	O	—	This pin is an output to control the gate of an external N-FET for voltage regulation.
VSS	P	—	Ground reference for logic and I/O pins.
VDD	P	—	Positive supply for logic and I/O pins.
$\overline{WR}$	I	TTL	Write control for parallel slave port (See $\overline{CS}$ and $\overline{RD}$ pins also).

Legend: TTL = TTL-compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
SM = SMBus compatible input. An external resistor is required if this pin is used as an output
NPU = N-channel pull-up
PU = Weak internal pull-up
No-P diode = No P-diode to VDD
AN = Analog input or output
I = input
O = output
P = Power
L = LCD Driver

4.5 Design Tips

No related design tips at this time.

4.6 Related Application Notes

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the Mid-Range MCU family (that is they may be written for the Base-Line, or High-End families), but the concepts are pertinent, and could be used (with modification and possible limitations). The current application notes related to Architecture are:

Title	Application Note #
No related application notes at this time.	

4.7 Revision History

Revision A

This is the initial released revision of the PICmicro's Architecture description.

Section 5. CPU and ALU

HIGHLIGHTS

This section of the manual contains the following major topics:

5.1 Introduction

The Central Processing Unit (CPU) is responsible for using the information in the program memory (instructions) to control the operation of the device. Many of these instructions operate on data memory. To operate on data memory, the Arithmetic Logical Unit (ALU) is required. In addition to performing arithmetical and logical operations, the ALU controls status bits (which are found in the STATUS register). The result of some instructions force status bits to a value depending on the state of the result.

The machine codes that the CPU recognizes are show in Table 5-1 (as well as the instruction mnemonics that the MPASM uses to generate these codes).

Table 5-1: Mid-Range MCU Instruction Set

Mnemonic, Operands		Description	Cycles	14-Bit Instruction Word				Status Bits Affected	Notes
				MSb			LSb		
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	Clear f	1	00	0001	lfff	ffff	Z	2
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000	lfff	ffff		
NOP	-	No Operation	1	00	0000	0xx0	0000		
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk		
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	-	Return from interrupt	2	00	0000	0000	1001		
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	-	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	-	Go into standby mode	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

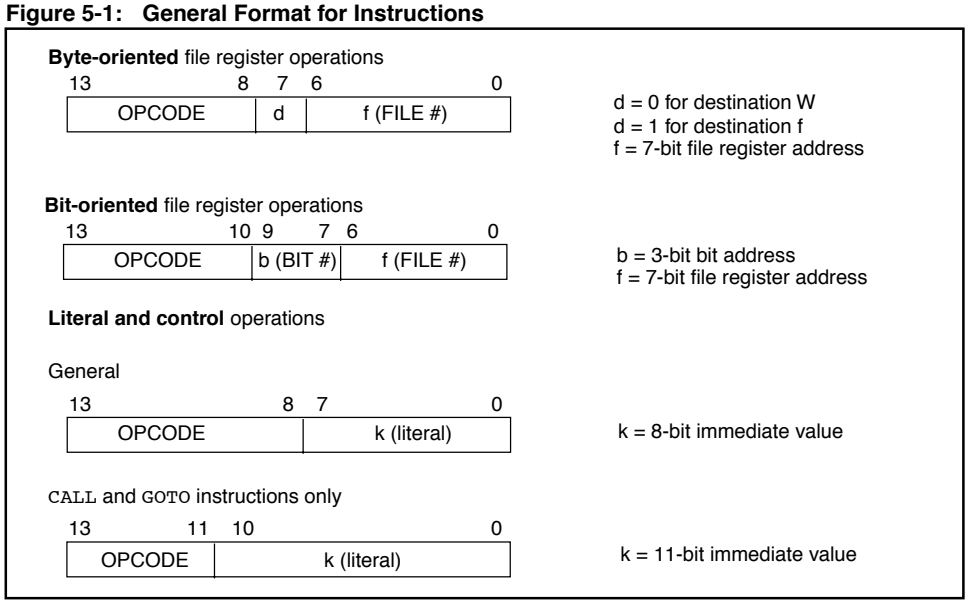
2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

5.2 General Instruction Format

The Mid-Range MCU instructions can be broken down into four general formats as shown in Figure 5-1. As can be seen the opcode for the instruction varies from 3-bits to 6-bits. This variable opcode size is what allows 35 instructions to be implemented.

Figure 5-1: General Format for Instructions



5.3 Central Processing Unit (CPU)

The CPU can be thought of as the “brains” of the device. It is responsible for fetching the correct instruction for execution, decoding that instruction, and then executing that instruction.

The CPU sometimes works in conjunction with the ALU to complete the execution of the instruction (in arithmetic and logical operations).

The CPU controls the program memory address bus, the data memory address bus, and accesses to the stack.

5.4 Instruction Clock

Each instruction cycle (TCY) is comprised of four Q cycles (Q1-Q4). The Q cycle time is the same as the device oscillator cycle time (TOSC). The Q cycles provide the timing/designation for the Decode, Read, Process Data, Write, etc., of each instruction cycle. The following diagram shows the relationship of the Q cycles to the instruction cycle.

The four Q cycles that make up an instruction cycle (TCY) can be generalized as:

Q1: Instruction Decode Cycle or forced No operation

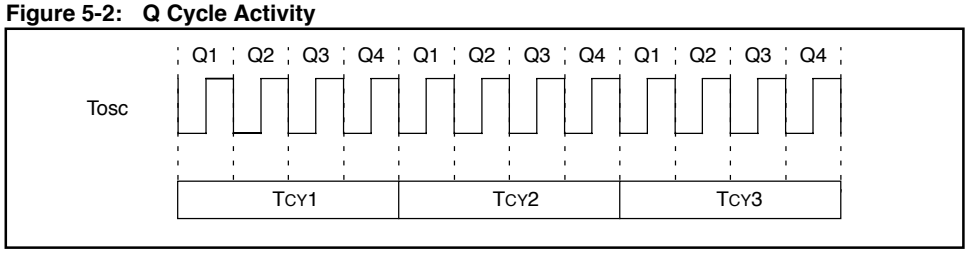
Q2: Instruction Read Data Cycle or No operation

Q3: Process the Data

Q4: Instruction Write Data Cycle or No operation

Each instruction will show a detailed Q cycle operation for the instruction.

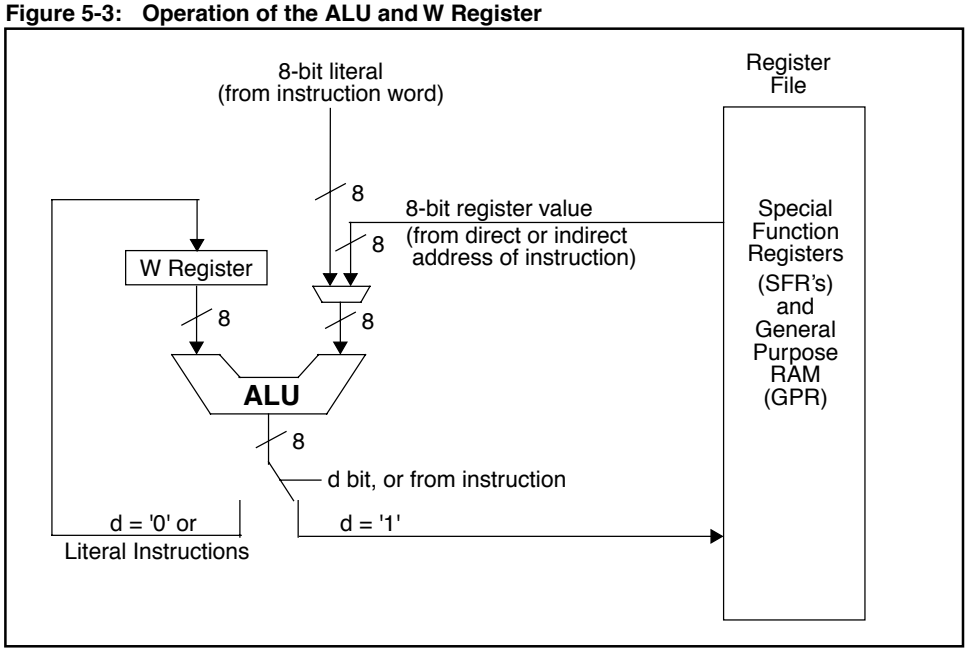
Figure 5-2: Q Cycle Activity



5.5 Arithmetic Logical Unit (ALU)

PICmicro MCUs contain an 8-bit ALU and an 8-bit working register. The ALU is a general purpose arithmetic and logical unit. It performs arithmetic and Boolean functions between the data in the working register and any register file.

Figure 5-3: Operation of the ALU and W Register



The ALU is 8-bits wide and is capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register). The other operand is a file register or an immediate constant. In single operand instructions, the operand is either the W register or a file register.

The W register is an 8-bit working register used for ALU operations. It is not an addressable register.

Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a borrow bit and a digit borrow out bit, respectively, in subtraction. See the SUBLW and SUBWF instructions for examples.

5.6 STATUS Register

The STATUS register, shown in Figure 5-1, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory. Since the selection of the Data Memory banks is controlled by this register, it is required to be present in every bank. Also, this register is in the same relative position (offset) in each bank (see **Figure 6-5: "Register File Map"** in the **"Memory Organization"** section).

The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper-three bits and set the Z bit. This leaves the STATUS register as `000u u1uu` (where `u` = unchanged).

It is recommended, therefore, that only `BCF, BSF, SWAPF` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect the Z, C or DC bits from the STATUS register. For other instructions, not affecting any status bits, see Table 5-1.

Note 1: Some devices do not require the IRP and RP1 (STATUS<7:6>) bits. These bits are not used by the Section 5. CPU and ALU and should be maintained clear. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward code compatibility with future products.

Note 2: The C and DC bits operate as a $\overline{\text{borrow}}$ and $\overline{\text{digit borrow}}$ bit, respectively, in subtraction.

Register 5-1: STATUS Register

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP:** Register Bank Select bit (used for indirect addressing)
1 = Bank 2, 3 (100h - 1FFh)
0 = Bank 0, 1 (00h - FFh)

For devices with only Bank0 and Bank1 the IRP bit is reserved, always maintain this bit clear.

bit 6:5 **RP1:RP0**: Register Bank Select bits (used for direct addressing)
11 = Bank 3 (180h - 1FFh)
10 = Bank 2 (100h - 17Fh)
01 = Bank 1 (80h - FFh)
00 = Bank 0 (00h - 7Fh)

Each bank is 128 bytes. For devices with only Bank0 and Bank1 the IRP bit is reserved, always maintain this bit clear.

bit 4 **$\overline{TO}$**: Time-out bit
1 = After power-up, CLRWDT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 3 **$\overline{PD}$**: Power-down bit
1 = After power-up or by the CLRWDT instruction
0 = By execution of the SLEEP instruction

bit2 **Z**: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC**: Digit carry/$\overline{\text{borrow}}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions) (for $\overline{\text{borrow}}$ the polarity is reversed)
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

bit 0 **C**: Carry/$\overline{\text{borrow}}$ bit (ADDWF, ADDLW,SUBLW,SUBWF instructions)
1 = A carry-out from the most significant bit of the result occurred
0 = No carry-out from the most significant bit of the result occurred

Note: For $\overline{\text{borrow}}$ the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (RRF, RLF) instructions, this bit is loaded with either the high or low order bit of the source register.

Legend		
R = Readable bit	W = Writable bit	
U = Unimplemented bit, read as '0'		- n = Value at POR reset

5.7 OPTION_REG Register

The OPTION_REG register is a readable and writable register which contains various control bits to configure the TMR0/WDT prescaler, the external INT Interrupt, TMR0, and the weak pull-ups on PORTB.

Register 5-2: OPTION_REG Register

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of INT pin
0 = Interrupt on falling edge of INT pin

bit 5 **T0CS**: TMR0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4 **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS2:PS0**: Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

Legend		
R = Readable bit	W = Writable bit	
U = Unimplemented bit, read as '0'		- n = Value at POR reset

Note: To achieve a 1:1 prescaler assignment for the TMR0 register, assign the prescaler to the Watchdog Timer.

5.8 PCON Register

The Power Control (PCON) register contains flag bit(s), that together with the TO and PD bits, allows the user to differentiate between the device resets.

Note 1:	$\overline{BOR}$ is unknown on Power-on Reset. It must then be set by the user and checked on subsequent resets to see if $\overline{BOR}$ is clear, indicating a brown-out has occurred. The $\overline{BOR}$ status bit is a don't care and is not necessarily predictable if the brown-out circuit is disabled (by clearing the BODEN bit in the Configuration word).
Note 2:	It is recommended that the $\overline{POR}$ bit be cleared after a power-on reset has been detected, so that subsequent power-on resets may be detected.

Register 5-3: PCON Register

R-u	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
MPEEN	—	—	—	—	$\overline{PER}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit 7 **MPEEN**: Memory Parity Error Circuitry Status bit
This bit reflects the value of the MPEEN configuration bit.

bit 6:3 **Unimplemented:** Read as '0'

bit 2 **$\overline{PER}$**: Memory Parity Error Reset Status bit
1 = No error occurred
0 = A program memory fetch parity error occurred
(must be set in software after a Power-on Reset occurs)

bit 1 **$\overline{POR}$**: Power-on Reset Status bit
1 = No Power-on Reset occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$**: Brown-out Reset Status bit
1 = No Brown-out Reset occurred
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Legend		
R = Readable bit	W = Writable bit	
U = Unimplemented bit, read as '0'		- n = Value at POR reset