CHAPTER 2

BRIEF DESCRIPTION OF THE
MICROCONTROLLER AND AMICUS18 IDE
USED IN THE PRESENT WORK

ABSTRACT

Because of the simple architecture and programming, PIC microcontrollers have become popular in the embedded systems. With this reason, brief architectural details of PIC 18F25K20 is explained in chapter 2. Also covered in this chapter are software details using Amicus 18 IDE.
2.1 Introduction:

PIC 18F25K20 is an 8-bit microcontroller from Microchip. Commercially available module with PIC 18F25K20 as the base microcontroller is used in the present work. This module is bought from NSK Electronics, Bangalore. It is popularly known as Amicus 18. The microcontroller in the Amicus 18 board is loaded with a boot loader. Amicus 18 board is flexible and easy-to-use for professionals and new comers in the field of embedded systems.

Thus Amicus 18 is an embedded system platform based on a simple open hardware design for a single-board microcontroller, with embedded I/O support and a standard programming language.

2.1.1 Key features of Amicus 18:

★ Micro chip's PIC 18F25K20
★ Can be programmed directly with USB cable.
★ Pre-loaded boot loader.
★ On board 3 and 5 volt linear regulators
★ Free development software (supported by crown hill, the developers of proton BASIC).
★ Not restricted to any one program language are (can use C18, Swordfish, etc.)
★ All pins of the controller are brought out to berg connector. Hence, when the USB connector is connected, the system (Laptop/Desktop) recognizes the host.
2.2 Salient features of PIC 18F25K20:

PIC 18F25K20 is available with 28/40/44-pin flash microcontrollers with nano watt XLP Technology. PIC 18F25K20 has the following features. Figure 2.1 and Figure 2.2 shows different pin packages of PIC 18F25K20 and PIC 18F45K20.

Figure 2.1 Different pin packages of PIC 18F25K20 and PIC 18F45K20
Figure 2.2 Different pin packages of PIC 18F25K20 and PIC 18F45K20

Note: R53 is the alternate pin for CCP2 multiplexing.
It has High-performance RISC CPU

- C Compiler Optimized Architecture:
  - Optional extended instruction set designed to optimize re-entrant code
- Up to 1024 bytes Data EEPROM
- Up to 64 Kbytes Linear Program Memory Addressing
- Up to 3936 bytes Linear Data Memory Addressing
- Up to 16 MIPS Operation
- 16-bit Wide Instructions, 8-bit Wide Data Path
- Priority Levels for Interrupts
- 31-Level, Software Accessible Hardware Stack

Flexible Oscillator Structure:

- Precision 16 MHz Internal Oscillator Block:
  - Factory calibrated to ± 1%
  - Software selectable frequencies range of 31 kHz to 16 MHz
  - 64 MHz performance available using PLL – No external components required
- Four Crystal modes up to 64 MHz
- Two External Clock modes up to 64 MHz
- 4X Phase Lock Loop (PLL)
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
- Allows for safe shutdown if peripheral clock stops
- Two-Speed Oscillator Start-up

→Special Microcontroller Features:

• Operating Voltage Range: 1.8V to 3.6V
• Self-Programmable under Software Control
• Programmable 16-Level High/Low-Voltage Detection (HLVD) module:
  - Interrupt on High/Low-Voltage Detection
• Programmable Brown-out Reset (BOR):
  - With software enable option
• Extended Watchdog Timer (WDT):
  - Programmable period from 4 ms to 131s
• Single-Supply 3V In-Circuit Serial Programming™ (ICSP™) via Two Pins
• In-Circuit Debug (ICD) via Two Pins

→Extreme Low-Power Management with NanoWatt XLP:

• Sleep mode: < 100 Na @ 1.8V
• Watchdog Timer: < 800 Na @ 1.8V
• Timer1 Oscillator: < 800 Na @ 32 kHz and 1.8V

→Analog Features:

• Analog-to-Digital Converter (ADC) module:
  - 10-bit resolution, 13 External Channels
  - Auto-acquisition capability
  - Conversion available during Sleep
- 1.2V Fixed Voltage Reference (FVR) channel
- Independent input multiplexing

• Analog Comparator module:
  - Two rail-to-rail analog comparators
  - Independent input multiplexing

• Voltage Reference (CVREF) module
  - Programmable (% VDD), 16 steps
  - Two 16-level voltage ranges using VREF pins

→Peripheral Highlights:

• Up to 35 I/O Pins plus 1 Input-only Pin:
  - High-Current Sink/Source 25 mA/25 mA
  - Three programmable external interrupts
  - Four programmable interrupt-on-change
  - Eight programmable weak pull-ups
  - Programmable slew rate

• Capture/Compare/PWM (CCP) module

• Enhanced CCP (ECCP) module:
  - One, two or four PWM outputs
  - Selectable polarity
  - Programmable dead time
  - Auto-Shutdown and Auto-Restart

• Master Synchronous Serial Port (MSSP) module
  - 3-wire SPI (supports all 4 modes)
  - I²C™ Master and Slave modes with address

Mask
• Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module:

- Supports RS-485, RS-232 and LIN
- RS-232 operation using internal oscillator
- Auto-Wake-up on Break
- Auto-Baud Detect

2.3 Architecture of PIC 18F25K20:

PIC microcontrollers are available in different packages namely: PDIP, SOIC, SSOP. PIC 18F25K20 device used in the present work comes in 40-pin PDIP package. Table 2.1 shows program memory, data memory, I/O pins, A/D converters, CCP/ECCP, PWM, SPI, I2C, EUSART, COMP and TIMER details of PIC 18F25K20 microcontroller. The pin assignment is shown in Figure 2.3, and Figure 2.4 shows the block diagram of PIC 18F25K20 microcontroller.

2.4.1 MEMORY ORGANIZATION

There are three types of memory in PIC18 Enhanced microcontroller devices:

• Program Memory
• Data RAM
• Data EEPROM

As Harvard architecture devices, the data and program memories use separate busses; this allows for concurrent access of the two memory spaces. The data EEPROM, for practical purposes, can be regarded as a peripheral device, since it is addressed and accessed through a set of control registers.
<table>
<thead>
<tr>
<th>Device</th>
<th>Program memory</th>
<th>Data memory</th>
<th>I/O</th>
<th>10-bit A/D</th>
<th>CCP/ECCP PWM</th>
<th>MSSP SPI Master</th>
<th>EUSART</th>
<th>COMP</th>
<th>TIMERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC 18F25K20</td>
<td>32K 16384</td>
<td>1536 256</td>
<td>25</td>
<td>11</td>
<td>1/1</td>
<td>Y Y</td>
<td>1</td>
<td>2</td>
<td>1/3</td>
</tr>
</tbody>
</table>

Table 2.1 Features of PIC 18F25K20
Figure 2.3 Pin diagram of PIC 18F25K20
Figure 2.4 PIC 18F25K20 (28-PIN) BLOCK DIAGRAM
2.4.1.1 Program Memory Organization

PIC18 microcontrollers implement a 21-bit program counter, which is capable of addressing a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all '0's (a NOP instruction).

These families of devices contain the following:

- PIC18F23K20, PIC18F43K20: 8 Kbytes of Flash Memory, up to 4,096 single-word
  Instructions
- PIC18F24K20, PIC18F44K20: 16 Kbytes of Flash Memory, up to 8,192 single-word
  instructions
- PIC18F25K20, PIC18F45K20: 32 Kbytes of Flash Memory, up to 16,384 single-word
  instructions
- PIC18F26K20, PIC18F46K20: 64 Kbytes of Flash Memory, up to 37,768 single-word
  instructions

PIC18 devices have two interrupt vectors. The Reset vector address is at 0000h and the interrupt vector addresses are at 0008h and 0018h.

2.4.1.2 PROGRAM COUNTER

The Program Counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21 bits wide and is contained in three separate 8-bit registers. The low byte, known as the PCL register, is both readable and writable. The high byte, or PCH register, contains the PC<15:8> bits; it is not directly readable or writable. Updates to the PCH register are performed through the PCLATH register. The upper byte is called PCU. This register contains the PC<20:16> bits; it is also not directly readable or writable. Updates to the PCU register are performed through the PCLATU register.
The contents of PCLATH and PCLATU are transferred to the program counter by any operation that writes PCL. Similarly, the upper two bytes of the program counter are transferred to PCLATH and PCLATU by an operation that reads PCL.

2.4.1.3 FLASH PROGRAM MEMORY

The Flash program memory is readable, writable and erasable during normal operation over the entire VDD range. A read from program memory is executed one byte at a time. A write to program memory is executed on blocks of 64, 32 or 16 bytes at a time depending on the specific device. Program memory is erased in blocks of 64 bytes at a time. The difference between the write and erase block sizes requires from 1 to 4 blocks write to restore the contents of a single block erase. A bulk erase operation cannot be issued from user code.

2.4.2 DATA EEPROM MEMORY

The data EEPROM is a nonvolatile memory array, separate from the data RAM and program memory, which is used for long-term storage of program data. It is not directly mapped in either the register file or program memory space but is indirectly addressed through the Special Function Registers (SFRs). The EEPROM is readable and writable during normal operation over the entire VDD range. Four SFRs are used to read and write to the data EEPROM as well as the program memory. They are:

- EECON1
- EECON2
- EEDATA
- EEADR
- EEADRH

The data EEPROM allows bytes read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and the EEADR: EEADRH register pair holds
the address of the EEPROM location being accessed. The EEPROM data memory is rated for high erase/write cycle endurance. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer; it will vary with voltage and temperature as well as from chip-to-chip.

2.5 Input/output Ports

PIC 18F25K20 has Four I/O ports. They are PORTA, PORTB, PORTC and PORTE.

Some pins of the I/O ports are multiplexed with an alternate function from the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

Each port has three registers for its operation. These registers are

- TRIS register (data direction register)
- PORT register (reads the levels on the pins of the devices)
- LAT register (output Latch)

The Data latch (LAT register) is useful for read-modify-write operations on the value that the I/O pins are driving.

PORTA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA. Setting a TRISA bit (=1) will make the corresponding PORTA pin an input. Clearing a TRISA bit (=0) will make the corresponding PORTA pin an output. The RA4 pin is multiplexed with the Timer0 module clock input and one of the comparator outputs to become the RA4/TOCK1/C1OUT pin.

Pins RA6 and RA7 are multiplexed with the main oscillator pins. The other PORTA pins are multiplexed with analog inputs. The operation of each pin is selected by ADCON1 register [1].
PORTB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB. PORTB is used as only simple I/O PORT in Amicus 18 Board.

PORTC Registers

PORTC is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISC. PORTC is multiplexed with several peripheral functions. The pins have Schmitt trigger input buffers. RC1 is the default configuration for the CCP2 peripheral pin. When enabling peripheral functions, care should be taken in defining TRIS bit for each PORTC pin. The EUSART and MSSP peripherals override the TRIS bit to make a pin an output or an input, depending on the peripheral configuration.

PORTE Registers

For PIC 18F25K20 devices, PORTE is only available when Master clear functionality is disabled (MCLR=0). In these cases PORTE is a single bit, input only port comprised of RE3 only.

2.6 Amicus 18 IDE

Proton Amicus18 is supported by an integrated development environment (Amicus IDE). The Amicus18 IDE has been designed to maximize programmer productivity by providing highly integrated and intuitive interface to the tools required to develop on the Amicus18 hardware.

The Amicus18 IDE provides many features for authoring, modifying, and compiling, deploying and debugging programmers'. The program can be compiled while being written, providing instant feedback on syntax errors. This results in an uninterrupted work flow from writing the program code through compiling to downloading the program to the hardware.
2.6.1 Procedure to use Amicus IDE

Install the Amicus 18 BASIC compiler which is around 50 Mb in size. Once installed, clicking on the Amicus 18 IDE, Icon is brought to the main program IDE. This is to write and edit the BASIC source code and start the compile process. Now it is ready to generate the machine HEX code for transferring to the PIC [2].

The Amicus IDE is divided into four main parts.

1. The menus and tools bar is at the top. 2. Below the toolbar there is code Explorer window at the left side. As we start to write the program, variables, labels and other definitions will appear on this window. 3. On right side there is large main code entry window where we enter the program source text. 4. There is a bottom section where status of information is shown. Figure 2.3 shows the Amicus18 IDE.

A couple of precautions are to be noted before start typing program. The compiler ignores blank lines. Normally they are inserted to make the source text easier to read; it's a bit like adding paragraphs to text.

Second, anything after a single quote is ignored by the compiler, add a standard comment block at the top of every program. This shows the name of the program, and date on which it is written.

Once entered the text it is always required to save program (Figure 2.4). By Pressing the "save" button from the toolbar, the IDE asks where would you like to store the file. Then store the files at the default location (or) any other place. Do not place them at the root of a drive. When the compile process starts it creates all sorts of additional files that soon clutter the place up.
Figure 2.3 Amicus IDE
Give the file name, for example program 1 and press the save button in the dialogue box. Next locate the “compile” button on the IDE toolbar (left hand side) and click it. Amicus will perform a sequence of activities including checking the text for obvious errors (Figure 2.5) and attempting to turn the statements to in PIC assembler (.ASM file). If the typed program is correct then a few seconds later the bottom of the IDE screen change to compilation success (Figure 2.6). If the program has mistakes, the compiler will attempt to tell where they are, and what it things the program is (Figure 2.7). Once finished writing the program, by hitting compile + program and the Amicus IDE will promptly have the PIC flashed (Figure 2.8).
Figure 2.4 Amicus IDE window to save the program
Figure 2.5 Errors display in the program
Figure 2.6 Amicus IDE with successful compilation
Figure 2.7 Errors display in the program
Figure 2.8 Amicus IDE window for loading the program on the target
REFERENCES


2. Amicus18 Help Menu from NSK Electronics, #66/3, Ground floor, Sabari complex, S.P.Road, Bangalore, KA, 560002.