PIC 18
Peripheral Interface Controller
Pramathesh Trivedi
Third Year Electronics Engineering, Atharva College of engineering, Mumbai, Maharashtra, India

Abstract: With the advancement in the field of micro processors and micro controllers, PIC 18 an invention of these millennia is proving to be one the promising controller. In the following context, some features of PIC 18 are highlighted.

Keywords: PIC 18, Memory organization, Pipelining, Instruction format, Addressing modes, RISC, Instruction set.

I. INTRODUCTION
Product of Microchip technology, PIC falls under family of Modified Harvard architecture Microcontroller. Controller being different from processors, due to inclusion of I/O, Memory and Processor in one single chip is compact and power saving. Controllers though not capable of fast processing as microprocessor but are widely used because of compact and power saving feature which is required in daily appliances, where processing is not issue but cost and power availability is. PICs are popular due to their wide availability, low cost, large user base, availability of low cost development tools. [1]

II. MEMORY ORGANIZATION
Due to availability of different on chip data memory and program memory, both are available at same for access.

As seen from diagram above, address bus to access program memory is 21 bit, thus can access at most 2 MB instructions. Since one memory location can store one byte information 21 bit equals $2^{21}$ locations = $2^{21}$ bytes = 2 MB

While instruction bus to access instructions is 16 bit long, reason being, almost all instructions of PIC 18 are 16 bit instructions, and thus having 16 bit instruction bus would ensure fetching of all instruction in one cycle.

For data memory, address bus provided is 12 bit long thus can access at most $2^{12}$ locations = $2^{12}$ bytes = 4096 bytes = 4 kB.
While corresponding data bus is 8 bit long, reason being PIC 18 can perform only 8 Bit operations.

![Fig.2. PIC 18 Data memory map [2]](image)

PIC18 data memory is implemented as SRAM. Every location of data memory is referred as register or file register. Due to condition applied on size of instruction, only 8 bit of address is specified in instruction. Thus can specify 256 addresses (2^8 bits = 256 addresses). So as to access whole memory, data memory is divided into 16 banks, each having 256 registers. Bank selection is possible with help of BSR register i.e. Bank select register. All special function registers i.e. SFRs are present at end i.e. Bank 15, so as to access SFRs along with general purpose registers, Access bank is used which has 128 general purpose registers from bank 0 which other 128 SFRs from Bank 15.

PIC 18 has 21 bit program memory counter, thus can access maximum 2 MB memory space. While stack in PIC is not part of Program memory, it is 31 entries each capable of storing 21 bits, present only to store return address of in case of interrupts and calls.

Address 0X000000 is assigned to reset vector, which is the starting address after power on. The address 0X000008 is assigned as starting address of ISR of high priority interrupt while address 0x000016 is starting address of ISR of low priority interrupt.

### III. PIPELINING

PIC being a true RISC processor executes perfect pipelining with less pipelining bubbles compared to a CISC processor. Pipelining is fetching of next instruction while current instruction is still being executed. This leads to better and speedy performance thus saving time. PIC performs a two stage pipelining i.e. it fetches instruction along with execution at same time which is performed very efficiently due to presence of different program and data memory and busses accessing them.

A detailed illustration of time saved due to pipelining is as follows.
Though pipelining is very beneficial, it too has some drawbacks namely: data dependency hazard and control hazard. Data dependency hazard occurs when result of current instruction acts as an operand for next instruction. Thus next instruction must be performed only after execution of current instruction. Or else value obtained would nothing but garbage value. Thus various algorithms are implemented to solve this problem.

Control hazard results due to Branch instructions. Because of pipelining next corresponding instruction is fetched for execution, but if current instruction happens to a branch instruction than program counter changes its value accordingly and next instruction which was already fetched is to be discarded leading to pipelining bubbles.

IV. INSTRUCTION FORMAT

There are five types of instruction format namely

- Byte-oriented operations
- Byte-to-Byte operations
- Bit-oriented file register operations
- Literal operation
- Control operations

Control operations

- Byte-oriented operations

Instruction is of 16 bit length, of which 8 bits are used to specify address of operand, one bit to specify which bank to be used, another bit to specify destination and remaining 6 bit to specify Opcode of the instruction.

\[ d=0 \text{ result to be stored in WREG register} \]
\[ d=1 \text{ result to be stored in file register} \]
\[ a=0 \text{ to force access bank} \]
\[ a=1 \text{ to use BSR to select bank} \]

- Byte-to-Byte operations

Only one instruction uses this format i.e. movff f1, f2

Instruction format is 32 bit long containing two 12 bit address and 4 bit long Opcode.

\[
\begin{array}{cccccc}
\text{Opcode} & \text{d} & \text{a} & \text{Address of operand} \\
\hline
15 & 12 & 11 & 0 \\
\end{array}
\]
• Bit-oriented file register operation
  This instruction format uses 8 bits to specify file register address, 3 bits to specify bit position, one bit to specify bank to be used, and remaining 4 bits to specify Opcode.

<table>
<thead>
<tr>
<th>15</th>
<th>12 11</th>
<th>9 8 7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opcode</td>
<td>b</td>
<td>a</td>
</tr>
</tbody>
</table>

• Literal Operations
  Instruction of this specify value of operand directly rather than their address. Such instructions have 8 bit operand value and 8 bit Opcode.

<table>
<thead>
<tr>
<th>15</th>
<th>8 7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opcode</td>
<td>Value of operand</td>
</tr>
</tbody>
</table>

• Control operations
  Control instructions are the branch instructions which include Goto (Unconditional long jump), Call, Bra (Unconditional short jump) and BC (Conditional Short jump).

Instruction format of Goto

<table>
<thead>
<tr>
<th>15</th>
<th>12 11</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opcode</td>
<td>Value of literal</td>
</tr>
</tbody>
</table>

Instruction format of call

<table>
<thead>
<tr>
<th>15</th>
<th>12 11</th>
<th>9 8 7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opcode</td>
<td>S</td>
<td>Value of literal</td>
</tr>
</tbody>
</table>

Instruction format of unconditional branch

<table>
<thead>
<tr>
<th>15</th>
<th>11 10</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opcode</td>
<td>Value of Literal</td>
</tr>
</tbody>
</table>

Instruction format of conditional branch

<table>
<thead>
<tr>
<th>15</th>
<th>8 7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opcode</td>
<td>Value of literal</td>
</tr>
</tbody>
</table>

V. ADDRESSING MODES

Manner in which operands are specified in an instruction is called addressing modes.

PIC 18 provides following addressing modes:

• Register direct:
  PIC 18 uses an 8-bit value to specify a data register as an operand. The register may be in access or other banks.
  Eg: movwf 0x0C, BANKED

• Immediate Mode:
  Actual operand is provided in the instruction, thus no need to access any memory location.
Eg: movlw 0x20

- **Inherent Mode:**
  In this mode operands are implied in Opcode field, thus instruction does not provide address of implied Opcode.
  Eg: addlw 0x81

- **Indirect Mode:**
  When a special function register is used to point to data memory than such addressing mode is called Indirect addressing mode.
  Eg: movwf INDF0

- **Bit direct addressing mode:**
  This instructions deal with bit level operations.
  Eg: BCF PORTB, 3, A

**VI. INSTRUCTION SET**

PIC 18 has 77 instructions of which only four instructions are 32 bit long while rest are 16 bit long. Complete instruction set is as follows. [4]
VII. CONCLUSION

With the benefits like parallel I/O ports, timer functions, C-Language friendly architecture, perfect RISC based architecture, perfect pipelining and compact but powerful instruction set; PIC 18 has proved to be ground shaking device. With the few faults of Intel micro controllers corrected PIC 18 not only shows better potential but also shows that its future in field of micro controllers is secured. Thus with all the benefits available at a cheap price PIC 18 has become favourite of hobbyist. A simple instruction set and seamless migration between product families make PIC microcontrollers the logical choice for design requiring flexibility and performance.

ACKNOWLEDGMENT

Information sourced from microchip technology’s datasheets.

REFERENCES