Chapter – 2

Functional description of microcontrollers and FPGA architecture
CHAPTER - 2

FUNCTIONAL DESCRIPTION OF MICROCONTROLLERS AND FPGA ARCHITECTURE

2.1. INTRODUCTION

The 8085 microprocessor is the base of all microprocessor [1]. It is used only for small applications like in toys, CROs etc., the first microprocessor used in the PC is 8086/8088 (16 bit microprocessor) [2]. Nowadays the microprocessors are used only for the general applications, and may not be used for specific applications. In place of microprocessors, microcontrollers can be used in developing many applications and easy to use and cost effective. Applications of microcontrollers also limited, due to its hardware design and cost. Advances in Embedded system design are System on Chips and Network on Chips. In the Chapter 3, the 8 bit microcontroller has been used to develop an embedded system for speed control vehicles nearby Educational Institutions. In the chapter 4, the advanced version of microcontrollers ARM LPC2148, a 32-bit microcontroller used for another embedded system application, that is designing the remote EB metering System. These two microcontrollers' features and functional architectures are briefly described in the following sections. The chapters 5, 6 and 7 are consists of the interesting area of the researcher in the field of Embedded system. In these chapters, the design and implementation of the work is based on FPGA. A brief description also presented in the following sections.
2.1.1. An Introduction Microprocessor

Computer is an electronic data processing machine which can perform the task as instructed in the program. It consists of five major functional units, they are:

1. Input unit
2. Output unit
3. ALU
4. Control unit
5. Memory

Input unit is used to feed the instructions to the processor to perform the particular task, there are many input devices, such as Keyboard, mouse, joystick etc.

Output unit is used to store the processed data permanently or to display the data.

ALU: It is the major unit able to process the given data, here only the various arithmetic and logical operations could be done.

CU: Control unit is used to issue the various control signals to the I/O devices or memory or any other associated devices, these signals are memory read, memory write or opcode fetch etc.

Memory: Before the execution of the program is stored in the memory and used to store the intermediate results. Memories are many types, used in the computer system depending on the requirements.

The combinations of the ALU and Control unit is known as CPU, once it was constructed using vacuum tubes, later using transistors, and FETs. They are large in size and occupies more space. With the invention of the microprocessor the CPU is shrunken to the small monolithic chip.
Microprocessor contains the following major sections or circuits:

1. ALU
2. Accumulator
3. Working registers / General purpose registers
4. Program counter
5. Stack pointer
6. Clock circuits and
7. Interrupt circuits

2.1.2. Introduction to the Microcontroller

A microcontroller is “true computer on a chip” [3] which contains all the features of the microprocessors (CPU). Apart from all these, it contains

1. ROM
2. RAM
3. Parallel I/O
4. Serial I/O
5. Counters / Timers
6. A/D converters and
7. Clock circuits

Hence it is called true computer on a chip, only minimum hardware are needed to make it functional. Whereas microprocessor is used in computer as CPU and many digital ICs are needed to get the full fledged system. Even though, microcontroller is a general purpose device, but it is used to control the operation of a machine using a fixed program that is stored in ROM, and that is not change over life time of the
system. This is known as embedded system. Nowadays there are many System on Chip Controller and Programmable System on Chips (PSOCs) are available, but microcontroller is the base of the embedded system.

### 2.1.3. Comparing Microprocessors and Microcontrollers

1. **µP** may have one or two types of bit handling instructions, where as microcontrollers will have many.
2. Microprocessors is concerned with rapid movement of code and data from external addresses to the chip.
3. **µC** is concerned with rapid movement of bits within the chip.
4. **µC** can function as a computer without the additional digital parts, such as parallel ports, serial ports, and interrupt controller etc.
5. **µP** must have many additional digital parts to be operational.

### 2.2. FEATURES OF 89S52 MICROCONTROLLER

The 89S52 microcontroller is the 8 bit microcontroller, has many features, and it can be used to develop small Embedded System applications. This microcontroller was chosen to develop for speed control of vehicles nearby Educational Institutions cited in the chapter 3 of this thesis. It is to study its advantages over 8 bit microprocessor.

- Compatible with MCS-51 products
- 8 KB of in-system-programmable (ISP) flash memory
- 4.0V to 5.5V operating range
- Fully static operation: 0Hz to 33 MHz
- Three level program memory lock
The AT89S52 [4] is a low-power, high-performance CMOS 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The device is manufactured using Atmel’s high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pin out. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications. The AT89S52 provides the standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S52 is designed with

- 256 × 8 bit internal RAM
- 32 Programmable I/O lines
- Three 16-bit Timer/Counters
- Eight interrupt sources
- Full duplex UART serial channel
- Low-power idle and power-down modes
- Interrupt Recovery from power down mode
- Watch dog timer
- Dual data pointer
- Power-off flag.

**Description**

The AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The device is manufactured using Atmel’s high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pin out. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications. The AT89S52 provides the standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S52 is designed with
static logic for operation down to zero frequency and supports two software selectable power saving modes. The idle mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset. Figure 2.1 shows the pin out details of 89S52 microcontroller.

![Pinout Diagram of 89S52 microcontroller](image_url)

**Figure 2.1: Pin out Diagram of 89S52 microcontroller**

**Pin Description**

- **VCC** - Supply voltage
- **GND** - Ground
Port 0

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high impedance inputs. Port 0 can also be configured to be the multiplexed low order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull ups. Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1

Port 1 is an 8-bit bidirectional I/O port with internal pull ups. The Port 1 output buffers can sink / source four Transistor Transistor Logic (TTL) inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pull ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current Integrated Injection Logic (IIL) because of the internal pull ups. In addition, P1.0 and P1.1 can be configured to be the timer / counter 2 external count input (P1.0/T2) and the timer / counter 2 trigger input (P1.1/T2EX), respectively, as shown in the following table. Port 1 also receives the low-order address bytes during Flash programming and verification. The Table 2.1 illustrates the alternate function of Port 1.

Table 2.1: Alternate Function of Port 1

<table>
<thead>
<tr>
<th>Port Pin</th>
<th>Alternate Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1.0</td>
<td>T2 (external count input to Timer/Counter 2), clock-out</td>
</tr>
<tr>
<td>P1.1</td>
<td>T2EX (Timer/Counter 2 capture/reload trigger and direction control)</td>
</tr>
<tr>
<td>P1.5</td>
<td>MOSI (used for In-System Programming)</td>
</tr>
<tr>
<td>P1.6</td>
<td>MISO (used for In-System Programming)</td>
</tr>
<tr>
<td>P1.7</td>
<td>SCK (used for In-System Programming)</td>
</tr>
</tbody>
</table>
Port 2

Port 2 is an 8-bit bidirectional I/O port with internal pull ups. The Port 2 output buffers can sink / source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the internal pull ups. Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register. Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pull ups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pull ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pull ups. Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table. Port 3 also receives some control signals for Flash programming and verification. The Table 2.2 Shows the alternate function of the Port 3.

Table 2.2: Alternate function of Port 3

<table>
<thead>
<tr>
<th>Port Pin</th>
<th>Alternate Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>P3.0</td>
<td>RXD (serial input port)</td>
</tr>
<tr>
<td>P3.1</td>
<td>TXD (serial output port)</td>
</tr>
<tr>
<td>P3.2</td>
<td>INT0 (external interrupt 0)</td>
</tr>
<tr>
<td>P3.3</td>
<td>INT1 (external interrupt 1)</td>
</tr>
<tr>
<td>P3.4</td>
<td>T0 (timer 0 external input)</td>
</tr>
<tr>
<td>P3.5</td>
<td>T1 (timer 1 external input)</td>
</tr>
<tr>
<td>P3.6</td>
<td>WR (external data memory write strobe)</td>
</tr>
<tr>
<td>P3.7</td>
<td>RD (external data memory read strobe)</td>
</tr>
</tbody>
</table>
RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 96 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

ALE/PROG

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming. In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory. If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction.

PSEN

Program Store Enable (PSEN) is the read strobe to external program memory. When the AT89S52 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

EA/VPP

External Access Enable. EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched
on reset. EA should be strapped to VCC for internal program executions. This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming.

**XTAL1**

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

**XTAL2**

It is the Output from the inverting oscillator amplifier.

The Figure 2.2 shows the block diagram of 89S52 microcontroller.
The various functional blocks of 89S52 microcontroller as follows:

**Special Function Registers**

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in Table 2.3.

### Table 2.3: AT89S52 SFR Map and Reset Values

<table>
<thead>
<tr>
<th>Address</th>
<th>Register</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>ACC</td>
<td>00000000</td>
</tr>
<tr>
<td>08H</td>
<td>B</td>
<td>00000000</td>
</tr>
<tr>
<td>09H</td>
<td>C</td>
<td>00000000</td>
</tr>
<tr>
<td>0Bh</td>
<td>D</td>
<td>00000000</td>
</tr>
<tr>
<td>0C8H</td>
<td>E</td>
<td>00000000</td>
</tr>
<tr>
<td>0C9H</td>
<td>F</td>
<td>00000000</td>
</tr>
<tr>
<td>0CBH</td>
<td>G</td>
<td>00000000</td>
</tr>
<tr>
<td>0CDH</td>
<td>H</td>
<td>00000000</td>
</tr>
<tr>
<td>0CFH</td>
<td>I</td>
<td>00000000</td>
</tr>
<tr>
<td>0D8H</td>
<td>J</td>
<td>00000000</td>
</tr>
<tr>
<td>0D9H</td>
<td>K</td>
<td>00000000</td>
</tr>
<tr>
<td>0DAH</td>
<td>L</td>
<td>00000000</td>
</tr>
<tr>
<td>0DBH</td>
<td>M</td>
<td>00000000</td>
</tr>
<tr>
<td>0DFH</td>
<td>N</td>
<td>00000000</td>
</tr>
<tr>
<td>0F0H</td>
<td>O</td>
<td>00000000</td>
</tr>
<tr>
<td>0F1H</td>
<td>P</td>
<td>00000000</td>
</tr>
<tr>
<td>0F2H</td>
<td>Q</td>
<td>00000000</td>
</tr>
<tr>
<td>0F3H</td>
<td>R</td>
<td>00000000</td>
</tr>
<tr>
<td>0F4H</td>
<td>S</td>
<td>00000000</td>
</tr>
<tr>
<td>0F5H</td>
<td>T</td>
<td>00000000</td>
</tr>
<tr>
<td>0F6H</td>
<td>U</td>
<td>00000000</td>
</tr>
<tr>
<td>0F7H</td>
<td>V</td>
<td>00000000</td>
</tr>
<tr>
<td>0F8H</td>
<td>W</td>
<td>00000000</td>
</tr>
<tr>
<td>0F9H</td>
<td>X</td>
<td>00000000</td>
</tr>
<tr>
<td>0FAH</td>
<td>Y</td>
<td>00000000</td>
</tr>
<tr>
<td>0FBH</td>
<td>Z</td>
<td>00000000</td>
</tr>
</tbody>
</table>

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect. User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.
Timer 2 Registers

Control and status bits are contained in registers T2CON (shown in Table 2.4) and T2MOD (shown in Table 2.5) for Timer 2. The register pair (RCAP2H, RCAP2L) are the Capture / Reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

Table 2.4: T2CON - Timer/Counter 2 Control Register

<table>
<thead>
<tr>
<th>Bit</th>
<th>TF2</th>
<th>EXF2</th>
<th>RCLK</th>
<th>TCLK</th>
<th>EXEN2</th>
<th>TR2</th>
<th>C/T2</th>
<th>CP/RL2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>0C8H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset Value</td>
<td>0000 0000B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit Addressable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.5: T2 MOD control register

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF2</td>
<td>Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1</td>
</tr>
<tr>
<td>EXF2</td>
<td>Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).</td>
</tr>
<tr>
<td>RCLK</td>
<td>Receive clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. TCLK = causes Timer 1 overflows to be used for the transmit clock.</td>
</tr>
<tr>
<td>TCLK</td>
<td>Transmit clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.</td>
</tr>
<tr>
<td>EXEN2</td>
<td>Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.</td>
</tr>
<tr>
<td>TR2</td>
<td>Start/Stop control for Timer 2. TR2 = 1 starts the timer</td>
</tr>
<tr>
<td>C/T2</td>
<td>Timer or counter select for Timer 2. C/T2 = 0 for timer function. C/T2 = 1 for external event counter (falling edge triggered).</td>
</tr>
<tr>
<td>CP/RL2</td>
<td>Capture / Reload select. CP/RL2 = 1 causes captures to occur on negative transitions at T2EX if EXEN2 = 1, CP/RL2 = 0 causes automatic reloads to occur when Timer 2 overflows or negative transitions occur at T2EX when EXEN2 = 1. When either RCLK or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.</td>
</tr>
</tbody>
</table>
**Interrupt Registers**

The individual interrupt enable bits are in the IE register. Two priorities can be set for each of the six interrupt sources in the IP register.

**Dual Data Pointer Registers**

To facilitate accessing both internal and external data memory, two banks of 16-bit Data Pointer Registers are provided: DP0 at SFR address locations 82H-83H and DP1 at 84H-85H. Bit DPS = 0 in SFR AUXR1 selects DP0 and DPS = 1 selects DP1. The user should always initialize the DPS bit to the appropriate value before accessing the respective Data Pointer Register.

**Power Off Flag**

The Power Off Flag (POF) is located at bit 4 (PCON.4) in the PCON SFR. POF is set to “1” during power up. It can be set and rest under software control and is not affected by reset.

**Memory Organization**

MCS-52 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

**Program Memory**

If the EA pin is connected to GND, all program fetches are directed to external memory. On the AT89S52, if EA is connected to VCC, program fetches to addresses 0000H through 1FFFH are directed to internal memory and fetches to addresses 2000H through FFFFH are to external memory.
Data Memory

The AT89S52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. This means that the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space. When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions which use direct addressing access of the SFR space. For example, the following direct addressing instruction accesses the SFR at location 0A0H (which is P2).

```
MOV OAOH, #data
```

Instructions that use indirect addressing access the upper 128 bytes of RAM. For example, the following indirect addressing instruction, where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

```
MOV @R0, #data
```

Note that stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space.

Watchdog Timer: (One-time Enabled with Reset-out)

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets. The WDT consists of a 13-bit counter and the Watchdog Timer Reset (WDTRST) SFR. The WDT is defaulted to disable from exiting reset. To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, it will increment every machine cycle while the oscillator is running.
The WDT timeout period is dependent on the external clock frequency. There is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET HIGH pulse at the RST pin.

**Using the WDT**

To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, the user needs to service it by writing 01EH and 0E1H to WDTRST to avoid a WDT overflow. The 13-bit counter overflows when it reaches 8191 (1FFFH), and this will reset the device. When the WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 8191 machine cycles. To reset the WDT the user must write 01EH and 0E1H to WDTRST.

WDTRST is a write-only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is $96 \times T_{OSC}$, where $T_{OSC} = 1/F_{OSC}$. To make the best use of the WDT, it should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

**WDT during Power-down and Idle**

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are two methods of exiting Power-down mode: by a hardware reset or via a level-activated external interrupt which is enabled prior to entering Power-down mode.
When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S52 is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode. To ensure that the WDT does not overflow within a few states of exiting Power-down, it is best to reset the WDT just before entering Power-down mode. Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S52 while in IDLE mode, the user should always set up a timer that will periodically exit IDLE, service the WDT, and reenter IDLE mode. With WDIDLE bit enabled, the WDT will stop to count in IDLE mode and resumes the count upon exit from IDLE.

UART

The UART in the AT89S52 operates the same way as the UART in the AT89C51.

Timer 0 and 1

Timer 0 and Timer 1 in the AT89S52 operate the same way as Timer 0 and Timer 1 in the AT89C51 and AT89C52.
Timer 2

Timer 2 is a 16-bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/T2 in the SFR T2CON, shown in Table 2.4. Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON, as shown in Table 2.6. Timer 2 consists of two 8-bit registers, TH2 and TL2. In the Timer function, the TL2 register is incremented every machine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

Table 2.6: Timer 2 Operating Modes

<table>
<thead>
<tr>
<th>RCLK + TCLK</th>
<th>CP / RL2</th>
<th>TR2</th>
<th>MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>16-bit Auto-reload</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>16-bit Capture</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>1</td>
<td>Baud Rate Generator</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>0</td>
<td>(Off)</td>
</tr>
</tbody>
</table>

In the Counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which the transition was detected. Since two machine cycles (24 oscillator periods) are required to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency. To ensure that a given level is sampled at least once before it changes, the level should be held for at least one full machine cycle.
Capture Mode

In the capture mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16-bit timer or counter which upon overflow sets bit TF2 in T2CON. This bit can then be used to generate an interrupt. If EXEN2 = 1, Timer 2 performs the same operation, but a 1 to 0 transition at external input T2EX also causes the current value in TH2 and TL2 to be captured into RCAP2H and RCAP2L, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. The EXF2 bit, like TF2, can generate an interrupt. The capture mode is illustrated in Figure 2.3.

![Figure 2.3: Timer 2 in capture mode](image)

Auto-reload (Up or Down Counter)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (see Table 2.5). Upon reset, the DCEN bit is set to 0 so that timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down, depending on the value of the T2EX pin.
Figure 2.4 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to OFFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H and RCAP2L. The values in Timer in Capture ModeRCAP2H and RCAP2L are preset by software. If EXEN2 = 1, a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at external input T2EX. This transition also sets the EXF2 bit. Both the TF2 and EXF2 bits can generate an interrupt if enabled.

Setting the DCEN bit enables Timer 2 to count up or down, as shown in Figure 2.5. In this mode, the T2EX pin controls the direction of the count.
A Logic 1 at T2EX makes Timer 2 count up. The timer will overflow at 0FFFFH and set the TF2 bit. This overflow also causes the 16-bit value in RCAP2H and RCAP2L to be reloaded into the timer registers, TH2 and TL2, respectively. A Logic 0 at T2EX makes Timer 2 count down. The timer underflows when TH2 and TL2 equal the values stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and causes 0FFFFH to be reloaded into the timer registers. The EXF2 bit toggles whenever Timer 2 overflows or underflows and can be used as a 17th bit of resolution. In this operating mode, EXF2 does not flag an interrupt.

**Table 2.7: Timer 2 Mode Control Register**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2OE</td>
<td>Timer 2 Output Enable bit</td>
</tr>
<tr>
<td>DCEN</td>
<td>When set, this bit allows Timer 2 to be configured as an up/down counter</td>
</tr>
</tbody>
</table>
Baud Rate Generator

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 2.4). Note that the baud rates for transmit and receive can be different if Timer 2 is used for the receiver or transmitter and Timer 1 is used for the other function. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 2.6.

![Figure 2.6: Timer 2 in Baud Rate Generator Mode](image)

The baud rate generator mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software. The baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate according to the following equation.

\[
\text{Modes 1 and 3 Baud Rates} = \frac{\text{Timer 2 Overflow Rate}}{16}
\]
The Timer can be configured for either timer or counter operation. In most applications, it is configured for timer operation (CP/T2 = 0). The timer operation is different for Timer 2 when it is used as a baud rate generator. Normally, as a timer, it increments every machine cycle (at 1/12 the oscillator frequency). As a baud rate generator, it increments every state time (at 1/2 the oscillator frequency). The baud rate formula is given below.

\[
\text{Modes 1 and 3 } = \frac{\text{Oscillator Frequency}}{32 \times [65536 - \text{RCAP2H, RCAP2L}]}
\]

where, (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer. Timer 2 as a baud rate generator is shown in Figure 2.6. This figure is valid only if RCLK or TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2 and will not generate an interrupt. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus, when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt. Note that when Timer 2 is running (TR2 = 1) as a timer in the baud rate generator mode, TH2 or TL2 should not be read from or written to. Under these conditions, the Timer is incremented every state time, and the results of a read or write may not be accurate. The RCAP2 registers may be read but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

**Programmable Clock Out**

A 50% duty cycle clock can be programmed to come out on P1.0, as shown in Figure 2.7.
This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed to input the external clock for Timer/Counter 2 or to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz at a 16 MHz operating frequency.

To configure the Timer/Counter 2 as a clock generator, bit C/T2 (T2CON.1) must be cleared and bit T20E (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer. The clock-out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L), as shown in the following equation.

\[
\text{Clock-out Frequency} = \frac{\text{Oscillator Frequency}}{4 \times 65536 - (\text{RCAP2H}, \text{RCAP2L})}
\]

In the clock-out mode, Timer 2 roll-overs will not generate an interrupt. This behavior is similar to when Timer 2 is used as a baud-rate generator. It is possible to
use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and clock-out frequencies cannot be determined independently from one another since they both use RCAP2H and RCAP2L.

**Interrupts**

The AT89S52 has a total of six interrupt vectors: two external interrupts (INT0 and INT1), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once. Note that Table 2.8 shows that bit position IE.6 is unimplemented. In the AT89S52, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future AT89 products. Timer 2 interrupt is generated by the logical OR of bits TF2 and EXF2 in register T2CON. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and that bit will have to be cleared in software. The Timer 0 and Timer 1 flags, TFO and TFI, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.

<table>
<thead>
<tr>
<th>(MSB)</th>
<th>(LSB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EA</td>
<td>-</td>
</tr>
<tr>
<td>ET2</td>
<td>ES</td>
</tr>
<tr>
<td>ET1</td>
<td>EX1</td>
</tr>
<tr>
<td>ET0</td>
<td>EX0</td>
</tr>
</tbody>
</table>

Enable Bit = 1 enables the interrupt.
Enable Bit = 0 disables the interrupt.
Table 2.8: Interrupt Enable (IE) Register

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Position</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>EA</td>
<td>IE.7</td>
<td>Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit</td>
</tr>
<tr>
<td>-</td>
<td>IE.6</td>
<td>Reserved</td>
</tr>
<tr>
<td>ET2</td>
<td>IE.5</td>
<td>Timer 2 interrupt enable bit</td>
</tr>
<tr>
<td>ES</td>
<td>IE.4</td>
<td>Serial Port interrupt enable bit</td>
</tr>
<tr>
<td>ET1</td>
<td>IE.3</td>
<td>Timer 1 interrupt enable bit</td>
</tr>
<tr>
<td>EX1</td>
<td>IE.2</td>
<td>External interrupt 1 enable bit</td>
</tr>
<tr>
<td>ET0</td>
<td>IE.1</td>
<td>Timer 0 interrupt enable bit</td>
</tr>
<tr>
<td>EX0</td>
<td>IE.0</td>
<td>External interrupt 0 enable bit</td>
</tr>
</tbody>
</table>

User software should never write 1s to unimplemented bits, because they may be used in future AT89 products

Oscillator Characteristics

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2.8.

![Oscillator Connections](image)

**Note:**

\[
C1, C2 = 30 \text{ pF} \pm 10 \text{ pF} \text{ for Crystals} \\
= 40 \text{ pF} \pm 10 \text{ pF} \text{ for Ceramic Resonators}
\]

Figure 2.8: Oscillator Connections
Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 2.9.

![Figure 2.9: External Clock Drive Configuration](image)

There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

**Idle Mode**

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset. Note that when idle mode is terminated by a hardware reset, the device normally resumes program execution from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when idle mode is terminated by a reset, the instruction following the one that invokes idle mode should not write to a port pin or to external memory.
Power-down Mode

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by an enabled external interrupt. Reset redefines the Special Function Registers (SFRs) but does not change the on-chip RAM. The reset should not be activated before VCC is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize. Table 2.9 illustrates the status of External Pins during Idle and Power-down Modes.

Table 2.9: Status of External Pins during Idle and Power-down Modes

<table>
<thead>
<tr>
<th>Mode</th>
<th>Program Memory</th>
<th>ALE</th>
<th>PSEN</th>
<th>PORT0</th>
<th>PORT1</th>
<th>PORT2</th>
<th>PORT3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle</td>
<td>Internal</td>
<td>1</td>
<td>1</td>
<td>Data</td>
<td>Data</td>
<td>Data</td>
<td>Data</td>
</tr>
<tr>
<td>Idle</td>
<td>External</td>
<td>1</td>
<td>1</td>
<td>Float</td>
<td>Data</td>
<td>Address</td>
<td>Data</td>
</tr>
<tr>
<td>Power-down</td>
<td>Internal</td>
<td>0</td>
<td>0</td>
<td>Data</td>
<td>Data</td>
<td>Data</td>
<td>Data</td>
</tr>
<tr>
<td>Power-down</td>
<td>External</td>
<td>0</td>
<td>0</td>
<td>Float</td>
<td>Data</td>
<td>Data</td>
<td>Data</td>
</tr>
</tbody>
</table>

2.3. LPC 2148 ARM MICROCONTROLLER

2.3.1. General description

The LPC2141/42/44/46/48 [5] microcontrollers are based on a 16-bit/32-bit ARM7TDMI-S CPU with real-time emulation and embedded trace support, that combine the microcontroller with embedded high-speed flash memory ranging from 32 kB to 512 kB. A 128-bit wide memory interface and a unique accelerator architecture enable 32-bit code execution at the maximum clock rate. For critical code size applications, the alternative 16-bit Thumb mode reduces code by more than 30%.
with minimal performance penalty. Due to their tiny size and low power consumption, LPC2141/42/44/46/48 are ideal for applications where miniaturization is a key requirement, such as access control and point-of-sale. Serial communications interfaces ranging from a USB 2.0 Full-speed device, multiple UARTs, SPI, SSP to I2C-bus and on-chip SRAM of 8 kB up to 40 kB, make these devices very well suited for communication gateways and protocol converters, soft modems, voice recognition and low end imaging, providing both large buffer size and high processing power. Various 32-bit timers, single or dual 10-bit ADC(s), 10-bit DAC, PWM channels and 45 fast GPIO lines with up to nine edge or level sensitive external interrupt pins make these microcontrollers suitable for industrial control and medical systems.

2.3.2. Features and benefits

- 16-bit/32-bit ARM7TDMI-S microcontroller in a tiny LQFP64 package.
- 8 kB to 40 kB of on-chip static RAM and 32 kB to 512 kB of on-chip flash memory.
- 128-bit wide interface / accelerator enables high-speed 60 MHz operation.
- In-System Programming / In-Application Programming (ISP/IAP) via on-chip boot loader software. Single flash sector or full chip erase in 400 ms and programming of 256 B in ms.
- Embedded ICE RT and Embedded Trace interfaces offer real-time debugging with then on-chip Real Monitor software and high-speed tracing of instruction execution.
- USB 2.0 Full-speed compliant device controller with 2 kB of endpoint RAM. In addition, the LPC2146/48 provides 8 kB of on-chip RAM accessible to USB by DMA.
• One or two (LPC2141/42 vs. LPC2144/46/48) 10-bit ADCs provide a total of 6/14 analog inputs, with conversion times as low as 2.44 µs per channel.
• Single 10-bit DAC provides variable analog output (LPC2142/44/46/48 only).
• Two 32-bit timers/external event counters (with four capture and four compare channels each), PWM unit (six outputs) and watchdog.
• Low power Real-Time Clock (RTC) with independent power and 32 kHz clock input. Multiple serial interfaces including two UARTs (16C550), two Fast I2C-bus (400 kbit/s), SPI and SSP with buffering and variable data length capabilities.
• Vectored Interrupt Controller (VIC) with configurable priorities and vector addresses.
• Up to 45 of 5 V tolerant fast general purpose I/O pins in a tiny LQFP64 package.
• Up to 21 external interrupt pins available.
• 60 MHz maximum CPU clock available from programmable on-chip PLL with settling time of 100 µs.
• On-chip integrated oscillator operates with an external crystal from 1 MHz to 25 MHz.
• Power saving modes include Idle and Power-down.
• Individual enable / disable of peripheral functions as well as peripheral clock scaling for additional power optimization.
• Processor wake-up from Power-down mode via external interrupt or BOD.
• Single power supply chip with POR and BOD circuits:
• CPU operating voltage range of 3.0 V to 3.6 V (3.3 V ± 10%) with 5 V tolerant I/O pads.
2.3.3. Architecture Overall View of LPC 2148

The Figure 2.10 shows the Block diagram illustration of LPC 2148 Microcontroller.

Figure 2.10: Block Diagram of LPC 2148
The ARM7TDMI-S is a general purpose 32-bit microprocessor, which offers high performance and very low power consumption. The ARM architecture is based on Reduced Instruction Set Computer (RISC) principles, and the instruction set and related decode mechanism are much simpler than those of micro-programmed Complex Instruction Set Computers (CISC). This simplicity results in a high instruction throughput and impressive real-time interrupt response from a small and cost-effective processor core. Pipeline techniques are employed so that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory. The ARM7TDMI-S processor also employs a unique architectural strategy known as Thumb, which makes it ideally suited to high-volume applications with memory restrictions, or applications where code density is an issue. The key idea behind Thumb is that of a super-reduced instruction set. Essentially, the ARM7TDMI-S processor has two instruction sets:

- The standard 32-bit ARM set.
- A 16-bit Thumb set.

The Thumb set’s 16-bit instruction length allows it to approach twice the density of standard ARM code while retaining most of the ARM’s performance advantage over a traditional 16-bit processor using 16-bit registers. This is possible because Thumb code operates on the same 32-bit register set as ARM code. Thumb code is able to provide up to 65% of the code size of ARM, and 160% of the performance of an equivalent ARM processor connected to a 16-bit memory system. The particular flash implementation in the LPC2141/42/44/46/48 allows for full speed execution also in ARM mode. It is recommended to program performance critical and
short code sections (such as interrupt service routines and DSP algorithms) in ARM mode. The impact on the overall code size will be minimal but the speed can be increased by 30% over Thumb mode.

2.3.4. On-Chip Flash Program Memory

The LPC2141/42/44/46/48 incorporate a 32 kB, 64 kB, 128 kB, 256 kB and 512 kB flash memory system respectively. This memory may be used for both code and data storage. Programming of the flash memory may be accomplished in several ways. It may be programmed In System via the serial port. The application program may also erase and/or program the flash while the application is running, allowing a great degree of flexibility for data storage field firmware upgrades, etc. Due to the architectural solution chosen for an on-chip boot loader, flash memory available for user's code on LPC2141/42/44/46/48 is 32 kB, 64 kB, 128 kB, 256 kB and 500 kB respectively. The LPC2141/42/44/46/48 flash memory provides a minimum of 100000 erase/write cycles and 20 years of data-retention.

2.3.5. On-Chip Static RAM

On-chip static RAM may be used for code and/or data storage. The Static Random Access Memory (SRAM) may be accessed as 8-bit, 16-bit, and 32-bit. The LPC2141, LPC2142/44 and LPC2146/48 provide 8 kB, 16 kB and 32 kB of static RAM respectively. In case of LPC2146/48 only, an 8 kB SRAM block intended to be utilized mainly by the USB can also be used as a general purpose RAM for data storage and code storage and execution.
2.3.6. Memory map

The LPC2141/42/44/46/48 memory map incorporates several distinct regions, as shown in Figure 2.11. In addition, the CPU interrupt vectors may be remapped to allow them to reside in either flash memory (the default) or on-chip static RAM.

<table>
<thead>
<tr>
<th>Memory Region</th>
<th>Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHB PERIPHERALS</td>
<td>0xFFFFFFFF</td>
</tr>
<tr>
<td>VPB PERIPHERALS</td>
<td>0xFFFFFFFF</td>
</tr>
<tr>
<td>RESERVED ADDRESS SPACE</td>
<td>0xFFFFFFFF</td>
</tr>
<tr>
<td>BOOT BLOCK (12 KB REMAPPED FROM ON-CHIP FLASH MEMORY)</td>
<td>0xFFFFFFFF</td>
</tr>
<tr>
<td>RESERVED ADDRESS SPACE</td>
<td>0xFFFFFFFF</td>
</tr>
<tr>
<td>8 KB ON-CHIP USB DMA RAM (LPC2146/2148)</td>
<td>0x7F60 0000</td>
</tr>
<tr>
<td>RESERVED ADDRESS SPACE</td>
<td>0x7F60 0000</td>
</tr>
<tr>
<td>32 KB ON-CHIP STATIC RAM (LPC2146/2148)</td>
<td>0x4000 0000</td>
</tr>
<tr>
<td>16 KB ON-CHIP STATIC RAM (LPC2142/2144)</td>
<td>0x0000 FFFF</td>
</tr>
<tr>
<td>8 KB ON-CHIP STATIC RAM (LPC2141)</td>
<td>0x0000 FFFF</td>
</tr>
<tr>
<td>RESERVED ADDRESS SPACE</td>
<td>0x0000 FFFF</td>
</tr>
<tr>
<td>TOTAL OF 512 KB ON-CHIP NON-VOLATILE MEMORY (LPC2148)</td>
<td>0x0000 0000</td>
</tr>
<tr>
<td>TOTAL OF 256 KB ON-CHIP NON-VOLATILE MEMORY (LPC2146)</td>
<td>0x0002 0000</td>
</tr>
<tr>
<td>TOTAL OF 128 KB ON-CHIP NON-VOLATILE MEMORY (LPC2144)</td>
<td>0x0001 0000</td>
</tr>
<tr>
<td>TOTAL OF 64 KB ON-CHIP NON-VOLATILE MEMORY (LPC2142)</td>
<td>0x0000 FFFF</td>
</tr>
<tr>
<td>TOTAL OF 32 KB ON-CHIP NON-VOLATILE MEMORY (LPC2141)</td>
<td>0x0000 0000</td>
</tr>
</tbody>
</table>

Figure 2.11: LPC2141/42/44/46/48 memory map
2.3.7. Interrupt controller

The Vectored Interrupt Controller (VIC) accepts all of the interrupt request inputs and categorizes them as Fast Interrupt ReQuest (FIQ), vectored Interrupt ReQuest (IRQ), and non vectored IRQ as defined by programmable settings. The programmable assignment scheme means that priorities of interrupts from the various peripherals can be dynamically assigned and adjusted. FIQ has the highest priority. If more than one request is assigned to FIQ, the VIC combines the requests to produce the FIQ signal to the ARM processor. The fastest possible FIQ latency is achieved when only one request is classified as FIQ, because then the FIQ service routine does not need to branch into the interrupt service routine but can run from the interrupt vector location. If more than one request is assigned to the FIQ class, the FIQ service routine will read a word from the VIC that identifies which FIQ source(s) is (are) requesting an interrupt. Vectored IRQs have the middle priority. Sixteen of the interrupt requests can be assigned to this category. Any of the interrupt requests can be assigned to any of the 16 vectored IRQ slots, among which slot 0 has the highest priority and slot 15 has the lowest. Non-vectored IRQs have the lowest priority. The VIC combines the requests from all the vectored and non-vectored IRQs to produce the IRQ signal to the ARM processor. The IRQ service routine can start by reading a register from the VIC and jumping there. If any of the vectored IRQs are pending, the VIC provides the address of the highest-priority requesting IRQs service routine, otherwise it provides the address of a default routine that is shared by all the non-vectored IRQs. The default routine can read another VIC register to see what IRQs are active.
2.3.7.1. Interrupt sources

Each peripheral device has one interrupt line connected to the Vectored Interrupt Controller, but may have several internal interrupt flags. Individual interrupt flags may also represent more than one interrupt source.

2.3.8. Pin connect block

The pin connect block allows selected pins of the microcontroller to have more than one function. Configuration registers control the multiplexers to allow connection between the pin and the on chip peripherals. Peripherals should be connected to the appropriate pins prior to being activated, and prior to any related interrupt(s) being enabled. Activity of any enabled peripheral function that is not mapped to a related pin should be considered undefined. The Pin Control Module with its pin select registers defines the functionality of the microcontroller in a given hardware environment. After reset all pins of Port 0 and Port 1 are configured as input with the following exceptions: If debug is enabled, the JTAG pins will assume their JTAG functionality, if trace is enabled, the Trace pins will assume their trace functionality. The pins associated with the I2C0 and I2C1 interface are open drain.

2.3.9. Fast general purpose parallel I/O (GPIO)

Device pins that are not connected to a specific peripheral function are controlled by the GPIO registers. Pins may be dynamically configured as inputs or outputs. Separate registers allow setting or clearing any number of outputs simultaneously. The value of the output register may be read back, as well as the current state of the port pins. LPC2141/42/44/46/48 introduce accelerated GPIO functions over prior LPC2000 devices:
• GPIO registers are relocated to the ARM local bus for the fastest possible I/O timing.
• Mask registers allow treating sets of port bits as a group, leaving other bits unchanged.
• All GPIO registers are byte addressable.
• Entire port value can be written in one instruction.

2.3.9.1. Features
• Bit-level set and clear registers allow a single instruction set or clear of any number of bits in one port.
• Direction control of individual bits.
• Separate control of output set and clear.
• All I/O default to inputs after reset.

2.3.10. 10-bit ADC

The LPC2141/42 contain one and the LPC2144/46/48 contain two analog to digital converters. These converters are single 10-bit successive approximation analog to digital converters. While ADC0 has six channels, ADC1 has eight channels. Therefore, total number of available ADC inputs for LPC2141/42 is 6 and for LPC2144/46/48 is 14.

2.3.10.1. Features
• 10 bit successive approximation analog to digital converter.
• Measurement range of 0 V to VREF (2.5 V ≤ VREF ≤ VDDA).
• Each converter capable of performing more than 400000 10-bit samples per second.
• Every analog input has a dedicated result register to reduce interrupt overhead.
• Burst conversion mode for single or multiple inputs.
• Optional conversion on transition on input pin or timer match signal.
• Global Start command for both converters (LPC2142/44/46/48 only).

2.3.11. 10-bit DAC

The DAC enables the LPC2141/42/44/46/48 to generate a variable analog output. The maximum DAC output voltage is the \( V_{\text{ref}} \) voltage.

2.3.11.1. Features

• 10-bit DAC.
• Buffered output.
• Power-down mode available.

2.3.12. USB 2.0 device controller

The USB is a 4-wire serial bus that supports communication between a host and a number (127 max) of peripherals. The host controller allocates the USB bandwidth to attached devices through a token based protocol. The bus supports hot plugging, unplugging, and dynamic configuration of the devices. All transactions are initiated by the host controller. The LPC2141/42/44/46/48 is equipped with a USB device controller that enables 12 Mbit/s data exchange with a USB host controller. It consists of a register interface, serial interface engine, endpoint buffer memory and DMA controller. The serial interface engine decodes the USB data stream and writes data to the appropriate end point buffer memory. The status of a completed USB transfer or error condition is indicated via status registers. An interrupt is also generated if enabled. A DMA controller (available in LPC2146/48 only) can transfer data between an endpoint buffer and the USB RAM.
2.3.12.1. Features

- Fully compliant with USB 2.0 Full-speed specification.
- Supports 32 physical (16 logical) endpoints.
- Supports control, bulk, interrupt and isochronous endpoints.
- Scalable realization of endpoints at run time.
- Endpoint maximum packet size selection (up to USB maximum specification) by software at run time.
- RAM message buffer size based on endpoint realization and maximum packet size.
- Supports SoftConnect and GoodLink LED indicator. These two functions are sharing one pin.
- Supports bus-powered capability with low suspend current.
- Supports DMA transfer on all non-control endpoints (LPC2146/48 only).
- One duplex DMA channel serves all endpoints (LPC2146/48 only).
- Allows dynamic switching between CPU controlled and DMA modes (only in LPC2146/48).
- Double buffer implementation for bulk and isochronous endpoints.

2.3.13. UARTs

The LPC2141/42/44/46/48 each contain two UARTs. In addition to standard transmit and receive data lines, the LPC2144/46/48 UART1 also provides a full modem control handshake interface. Compared to previous LPC2000 microcontrollers, UARTs in LPC2141/42/44/46/48 introduce a fractional baud rate generator for both UARTs, enabling these microcontrollers to achieve standard baud rates such as
115200 with any crystal frequency above 2 MHz. In addition, auto-CTS/RTS flow-control functions are fully implemented in hardware (UART1 in LPC2144/46/48 only).

2.3.13.1. Features

- 16 B Receive and Transmit FIFOs.
- Register locations conform to 16C550 industry standard.
- Receiver FIFO trigger points at 1 B, 4 B, 8 B, and 14 B
- Built-in fractional baud rate generator covering wide range of baud rates without a need for external crystals of particular values.
- Transmission FIFO control enables implementation of software (XON/XOFF) flow control on both UARTs.
- LPC2144/46/48 UART1 equipped with standard modem interface signals. This module also provides full support for hardware flow control (auto-CTS/RTS).

2.3.14. I2C-bus serial I/O controller

The LPC2141/42/44/46/48 each contain two I2C-bus controllers. The I2C-bus is bidirectional, for inter-IC control using only two wires: a Serial Clock Line (SCL), and a Serial Data Line (SDL). Each device is recognized by a unique address and can operate as either a receiver-only device (e.g., an LCD driver or a transmitter with the capability to both receive and send information (such as memory)). Transmitters and/or receivers can operate in either master or slave mode, depending on whether the chip has to initiate a data transfer or is only addressed. The I2C-bus is a multi-master
bus, it can be controlled by more than one bus master connected to it. The I2C-bus implemented in LPC2141/42/44/46/48 supports bit rates up to 400 kbit/s (Fast I2C-bus).

2.3.14.1. Features

- Compliant with standard I2C-bus interface.
- Easy to configure as master, slave, or master/slave.
- Programmable clocks allow versatile rate control.
- Bidirectional data transfer between masters and slaves.
- Multi-master bus (no central master).
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus.
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus.
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer.
- The I2C-bus can be used for test and diagnostic purposes.

2.3.15. SPI serial I/O controller

The LPC2141/42/44/46/48 each contain one SPI controller. The SPI is a full duplex serial interface, designed to handle multiple masters and slaves connected to a given bus. Only a single master and a single slave can communicate on the interface during a given data transfer. During a data transfer the master always sends a byte of data to the slave, and the slave always sends a byte of data to the master.
2.3.15.1. Features

- Compliant with SPI specification.
- Synchronous, Serial, Full Duplex, Communication.
- Combined SPI master and slave.
- Maximum data bit rate of one eighth of the input clock rate.

2.3.16. SSP serial I/O controller

The LPC2141/42/44/46/48 each contain one Serial Synchronous Port controller (SSP). The SSP controller is capable of operation on a SPI, 4-wire SSI, or Micro-wire bus. It can interact with multiple masters and slaves on the bus. However, only a single master and a single slave can communicate on the bus during a given data transfer. The SSP supports full duplex transfers, with data frames of 4 bits to 16 bits of data flowing from the master to the slave and from the slave to the master. Often only one of these data flows carries meaningful data.

2.3.16.1 Features

- Compatible with Motorola’s SPI, TI’s 4-wire SSI and National Semiconductor’s.
- Micro-wire buses.
- Synchronous serial communication.
- Master or slave operation.
- 8-frame FIFOs for both transmit and receive.
- Four bits to 16 bits per frame.

2.3.17. General purpose timers/external event counters

The Timer/Counter is designed to count cycles of the Peripheral Clock (PCLK) or an externally supplied clock and optionally generate interrupts or perform
other actions at specified timer values, based on four match registers. It also includes four capture inputs to trap the timer value when an input signal transitions, optionally generating an interrupt. Multiple pins can be selected to perform a single capture or match function, providing an application with ‘or’ and ‘and’, as well as ‘broadcast’ functions among them. The LPC2141/42/44/46/48 can count external events on one of the capture inputs if the minimum external pulse is equal or longer than a period of the PCLK. In this configuration, unused capture lines can be selected as regular timer capture inputs, or used as external interrupts.

2.3.17.1. Features

- A 32-bit timer / counter with a programmable 32-bit prescaler.
- External event counter or timer operation.
- Four 32-bit capture channels per timer / counter that can take a snapshot of the timer value when an input signal transitions. A capture event may also optionally generate an interrupt.
- Four 32-bit match registers that allow:
  - Continuous operation with optional interrupt generation on match.
  - Stop timer on match with optional interrupt generation.
  - Reset timer on match with optional interrupt generation.
- Four external outputs per timer / counter corresponding to match registers, with the following capabilities:
  - Set LOW on match.
  - Set HIGH on match.
  - Toggle on match.
  - Do nothing on match.
2.3.18. Watchdog timer

The purpose of the watchdog is to reset the microcontroller within a reasonable amount of time if it enters an erroneous state. When enabled, the watchdog will generate a system reset if the user program fails to 'feed' (or reload) the watchdog within a predetermined amount of time.

2.3.18.1. Features

- Internally resets chip if not periodically reloaded.
- Debug mode.
- Enabled by software but requires a hardware reset or a watchdog reset/interrupt to be disabled.
- Incorrect/Incomplete feed sequence causes reset/interrupt if enabled.
- Flag to indicate watchdog reset.
- Programmable 32-bit timer with internal pre-scaler.
- Selectable time period from \((T_{cy}(PCLK) \times 256 \times 4)\) to \((T_{cy}(PCLK) \times 232 \times 4)\) in multiples of \(T_{cy}(PCLK) \times 4\).

2.3.19. Real-time clock

The RTC is designed to provide a set of counters to measure time when normal or idle operating mode is selected. The RTC has been designed to use little power, making it suitable for battery powered systems where the CPU is not running continuously (Idle mode).

2.3.19.1. Features

- Measures the passage of time to maintain a calendar and clock.
- Ultra-low power design to support battery powered systems.
• Provides Seconds, Minutes, Hours, Day of Month, Month, Year, Day of Week, and Day of Year.

• Can use either the RTC dedicated 32 kHz oscillator input or clock derived from the external crystal/oscillator input at XTAL1. Programmable reference clock divider allows fine adjustment of the RTC.

• Dedicated power supply pin can be connected to a battery or the main 3.3 V.

2.3.20. Pulse width modulator

The PWM is based on the standard timer block and inherits all of its features, although only the PWM function is pinned out on the LPC2141/42/44/46/48. The timer is designed to count cycles of the Peripheral Clock (PCLK) and optionally generate interrupts or perform other actions when specified timer values occur, based on seven match registers. The PWM function is also based on match register events. The ability to separately control rising and falling edge locations allows the PWM to be used for more applications. For instance, multi-phase motor control typically requires three non-overlapping PWM outputs with individual control of all three pulse widths and positions. Two match registers can be used to provide a single edge controlled PWM output. One match register (MRO) controls the PWM cycle rate, by resetting the count upon match. The other match register controls the PWM edge position. Additional single edge controlled PWM outputs require only one match register each, since the repetition rate is the same for all PWM outputs. Multiple single edge controlled PWM outputs will all have a rising edge at the beginning of each PWM cycle, when an MRO match occurs. Three match registers can be used to provide a PWM output with both edges controlled. Again, the MRO match register
controls the PWM cycle rate. The other match registers control the two PWM edge positions. Additional double edge controlled PWM outputs require only two match registers each, since the repetition rate is the same for all PWM outputs. With double edge controlled PWM outputs, specific match registers control the rising and falling edge of the output. This allows both positive going PWM pulses (when the rising edge occurs prior to the falling edge), and negative going PWM pulses (when the falling edge occurs prior to the rising edge).

2.3.20.1. Features

- Seven match registers allow up to six single edge controlled or three double edge controlled PWM outputs, or a mix of both types.

- The match registers also allow:
  - Continuous operation with optional interrupt generation on match.
  - Stop timer on match with optional interrupt generation.
  - Reset timer on match with optional interrupt generation.

- Supports single edge controlled and/or double edge controlled PWM outputs.
  Single edge controlled PWM outputs all go HIGH at the beginning of each cycle unless the output is a constant LOW. Double edge controlled PWM outputs can have either edge occur at any position within a cycle. This allows for both positive going and negative going pulses.

- Pulse period and width can be any number of timer counts. This allows complete flexibility in the trade-off between resolution and repetition rate. All PWM outputs will occur at the same repetition rate.

- Double edge controlled PWM outputs can be programmed to be either positive going or negative going pulses.
- Match register updates are synchronized with pulse outputs to prevent generation of erroneous pulses. Software must ‘release’ new match values before they can become effective.
- May be used as a standard timer if the PWM mode is not enabled.
- A 32-bit Timer/Counter with a programmable 32-bit Prescaler.

2.3.21. System Control
2.3.21.1. Crystal Oscillator

On-chip integrated oscillator operates with external crystal in range of 1 MHz to 25 MHz. The oscillator output frequency is called fosc and the ARM processor clock frequency is referred to as CCLK for purposes of rate equations, etc. fosc and CCLK are the same value unless the PLL is running and connected.

2.3.21.2. Phase Lock Loop (PLL)

The PLL accepts an input clock frequency in the range of 10 MHz to 25 MHz. The input frequency is multiplied up into the range of 10 MHz to 60 MHz with a Current Controlled Oscillator (CCO). The multiplier can be an integer value from 1 to 32 (in practice, the multiplier value cannot be higher than 6 on this family of microcontrollers due to the upper frequency limit of the CPU). The CCO operates in the range of 156 MHz to 320 MHz, so there is an additional divider in the loop to keep the CCO within its frequency range while the PLL is providing the desired output frequency. The output divider may be set to divide by 2, 4, 8, or 16 to produce the output clock. Since the minimum output divider value is 2, it is insured that the PLL output has a 50% duty cycle. The PLL is turned off and bypassed following a chip reset and may be enabled by software. The program must configure and activate...
the PLL, wait for the PLL to Lock, then connect to the PLL as a clock source. The PLL settling time is 100 µs.

2.3.21.3. Reset and wake-up timer

Reset has two sources on the LPC2141/42/44/46/48: the RESET pin and watchdog reset. The RESET pin is a Schmitt trigger input pin with an additional glitch filter. Assertion of chip reset by any source starts the Wake-up Timer (see Wake-up Timer description below), causing the internal chip reset to remain asserted until the external reset is de-asserted, the oscillator is running, a fixed number of clocks have passed, and the on-chip flash controller has completed its initialization. When the internal reset is removed, the processor begins executing at address 0, which is the reset vector. At that point, all of the processor and peripheral registers have been initialized to predetermined values. The Wake-up Timer ensures that the oscillator and other analog functions required for chip operation are fully functional before the processor is allowed to execute instructions. This is important at power on, all types of reset, and whenever any of the aforementioned functions are turned off for any reason. Since the oscillator and other functions are turned off during Power-down mode, any wake-up of the processor from Power-down mode makes use of the Wake-up Timer. The Wake-up Timer monitors the crystal oscillator as the means of checking whether it is safe to begin code execution. When power is applied to the chip, or some event caused the chip to exit Power-down mode, some time is required for the oscillator to produce a signal of sufficient amplitude to drive the clock logic. The amount of time depends on many factors, including the rate of VDD ramp (in the case of power on), the type of crystal and its electrical characteristics (if a quartz
crystal is used), as well as any other external circuitry (e.g. capacitors), and the characteristics of the oscillator itself under the existing ambient conditions.

2.3.21.4. Brownout detector

The LPC2141/42/44/46/48 include 2-stage monitoring of the voltage on the VDD pins. If this voltage falls below 2.9 V, the BOD asserts an interrupt signal to the VIC. This signal can be enabled for interrupt, if not, software can monitor the signal by reading dedicated register. The second stage of low voltage detection asserts reset to inactivate the LPC2141/42/44/46/48 when the voltage on the VDD pins falls below 2.6 V. This reset prevents alteration of the flash as operation of the various elements of the chip would otherwise become unreliable due to low voltage. The BOD circuit maintains this reset down below 1 V, at which point the POR circuitry maintains the overall reset. Both the 2.9 V and 2.6 V thresholds include some hysteresis. In normal operation, this hysteresis allows the 2.9 V detection to reliably interrupt, or a regularly-executed event loop to sense the condition.

2.3.21.5. Code Security

This feature of the LPC2141/42/44/46/48 allows an application to control whether it can be debugged or protected from observation. If after reset on-chip boot loader detects a valid checksum in flash and reads 0x8765 4321 from address 0x1FC in flash, debugging will be disabled and thus the code in flash will be protected from observation. Once debugging is disabled, it can be enabled only by performing a full chip erase using the ISP.
2.3.21.6. External Interrupt Inputs

The LPC2141/42/44/46/48 include up to nine edge or level sensitive External Interrupt Inputs as selectable pin functions. When the pins are combined, external events can be processed as four independent interrupt signals. The External Interrupt Inputs can optionally be used to wake-up the processor from Power-down mode. Additionally capture input pins can also be used as external interrupts without the option to wake the device up from Power-down mode.

2.3.21.7. Memory mapping control

The Memory Mapping Control alters the mapping of the interrupt vectors that appear beginning at address 0x0000 0000. Vectors may be mapped to the bottom of the on-chip flash memory, or to the on-chip static RAM. This allows code running in different memory spaces to have control of the interrupts.

2.3.21.8. Power control

The LPC2141/42/44/46/48 supports two reduced power modes: Idle mode and Power-down mode. In Idle mode, execution of instructions is suspended until either a reset or interrupt occurs.

Peripheral functions continue operation during idle mode and may generate interrupts to cause the processor to resume execution. Idle mode eliminates power used by the processor itself, memory systems and related controllers, and internal buses. In Power-down mode, the oscillator is shut down and the chip receives no internal clocks. The processor state and registers, peripheral registers, and internal SRAM values are preserved throughout Power-down mode and the logic levels of chip output pins remain static. The Power-down mode can be terminated and normal
operation resumed by either a reset or certain specific interrupts that are able to function without clocks. Since all dynamic operation of the chip is suspended, Power-down mode reduces chip power consumption to nearly zero. Selecting an external 32 kHz clock instead of the PCLK as a clock-source for the on-chip RTC will enable the microcontroller to have the RTC active during Power-down mode. Power-down current is increased with RTC active. However, it is significantly lower than in idle mode. A Power Control for Peripherals feature allows individual peripherals to be turned off if they are not needed in the application, resulting in additional power savings during active and idle mode.

2.4. FPGA ARCHITECTURE

FPGA (Field Programmable Gate Array) is an integrated circuit containing gate matrix which can be programmed by the user “in the field” without using expensive equipment. An FPGA contains a set of programmable logic gates and rich interconnect resources, making it possible to implement complex digital circuits. FPGA devices are produced by a number of semiconductor companies: Xilinx, Altera, Actel, Lattice, QuickLogic and Atmel.

2.4.1. FPGA Implementation Technologies

Configuration bit stream can be stored in FPGA using various technologies. The majority of FPGAs is based on SRAM (Static RAM).

2.4.1.1. SRAM-based FPGAs

SRAM-based FPGA [6] stores logic cells configuration data in the static memory (organized as an array of latches). An example of usage of SRAM-controlled
switches is illustrated in Figure 2.12, showing two applications of SRAM cells: for controlling the gate nodes of pass-transistor switches and to control the select lines of multiplexers that drive logic block inputs.

![Logic Cell](image)

**Figure 2.12: SRAM based FPGA**

The figure gives an example of the connection of one logic block (represented by the AND-gate in the upper left corner) to another through two pass-transistor switches, and then a multiplexer, all controlled by SRAM cells. Whether an FPGA uses pass-transistors or multiplexers or both depends on the particular product. Since SRAM is volatile and can’t keep data without power source, such FPGAs must be programmed (configured) upon start. There are two basic modes of programming.

- **Master mode**, when FPGA reads configuration data from an external source, such as an external Flash memory chip.
- **Slave mode**, when FPGA is configured by an external master device, such as a processor. This can be usually done via a dedicated configuration interface or via a boundary-scan (JTAG) interface.
SRAM-based FPGAs include most chips of Xilinx Virtex and Spartan families and Altera Stratix and Cyclone. The Figure 2.13 shows the single SRAM memory cell.

2.4.1.2. SRAM-based FPGAs with an Internal Flash Memory

This type of FPGA is generally like the previous, except that these chips contain internal flash memory blocks, thus eliminating the need to have an external non-volatile memory.

The true flash-based FPGAs shouldn’t be confused with the previous type. The SRAM-based FPGAs with internal flash memory use flash only during startup to load data to the SRAM configuration cells. On the contrary, true flash-based FPGA uses flash as a primary resource for configuration storage, and doesn’t require SRAM.
(a similar technology is used in CPLDs - Complex Programmable Logic Devices, but the FPGA architecture is very different from that of CPLD). This technology has an advantage of being less power consumptive. Flash-based FPGAs are also more tolerant to radiation effects. Figure 2.14 shows the illustration of a floating gate transistor used in flash memory.

2.4.2. Antifuse-based FPGAs

Anti-fuse-based FPGAs are different from the previous ones in that they can be programmed only once. The anti-fuse is a device that doesn't conduct current initially, but can be “burned” to conduct current (the anti-fuse behavior is thus opposite to that of the fuse, hence the name). The anti-fuse-based FPGA can't be then reprogrammed since there is no way to return a burned antifuse into the initial state. Anti-fuses are suitable for FPGAs because they can be built using modified CMOS technology. As an example, Actel’s antifuse structure, known as PLICE, is depicted in Figure 2.15. The figure shows that an antifuse is positioned between two interconnect wires and physically consists of three sandwiched layers, the top and bottom layers are conductors, and the middle layer is an insulator. When unprogrammed, the insulator isolates the top and bottom layers, but when programmed the insulator changes to become a low-resistance link.

![Figure 2.15: Actel Antifuse Structure](image-url)
Modern SRAM-based FPGAs have highest densities, but consume a lot of power and need an external non-volatile memory to store configuration bit-stream. SRAM-based FPGAs with an internal flash module doesn’t need an external configuration memory. Flash-based and Antifuse-based FPGAs consume much less power than their SRAM-based counterparts. Antifuse-based FPGAs can only be programmed once.

2.5. NIOS II SYSTEM

Altera’s NIOS II is a soft processor [7], defined in a hardware description language, which can be implemented in Altera’s FPGA devices by using the Quartus II CAD system. The NIOS II processor can be used with a variety of other components to form a complete system. These components include a number of standard peripherals, but it is also possible to define custom peripherals. Altera’s DE2 Development and Education board contains several components that can be integrated into a NIOS II system. An example of such a system is shown in Figure 2.16.

![NIOS II system diagram](image-url)

**Figure 2.16:** A NIOS II system implemented on the DE2 board
The Nios II processor and the interfaces needed to connect to other chips on the DE2 board are implemented in the Cyclone II FPGA chip. These components are interconnected by means of the interconnection network called the Avalon Switch Fabric. Memory blocks in the Cyclone II device can be used to provide an on-chip memory for the Nios II processor. They can be connected to the processor either directly or through the Avalon network. The SRAM and SDRAM memory chips on the DE2 board are accessed through the appropriate interfaces. Input/output interfaces are instantiated to provide connection to the I/O devices used in the system. A special JTAG UART interface is used to connect to the circuitry that provides a Universal Serial Bus (USB) link to the host computer to which the DE2 board is connected. This circuitry and the associated software is called the USB-Blaster. Another module, called the JTAG Debug module, is provided to allow the host computer to control the Nios II processor. It makes it possible to perform operations such as downloading programs into memory, starting and stopping execution, setting program breakpoints, and collecting real-time execution trace data.

Since all parts of the NIOS II system implemented on the FPGA chip are defined by using a hardware description language, a knowledgeable user could write such code to implement any part of the system. This would be an onerous and time consuming task. Instead, one can use the SOPC Builder tool in the Quartus II software to implement a desired system simply by choosing the required components and specifying the parameters needed to make each component fit the overall requirements of the system.
2.5.1. Overview of NIOS II Processor Features

The NIOS II processor has a number of features that can be configured by the user to meet the demands of a desired system. The processor can be implemented in three different configurations:

- NIOS II/f is a "fast" version designed for superior performance. It has the widest scope of configuration options that can be used to optimize the processor for performance.
- NIOS II/s is a "standard" version that requires less resources in an FPGA device as a trade-off for reduced performance.
- NIOS II/e is an "economy" version which requires the least amount of FPGA resources, but also has the most limited set of user-configurable features.

The NIOS II processor has a Reduced Instruction Set Computer (RISC) architecture. Its arithmetic and logic operations are performed on operands in the general purpose registers. The data is moved between the memory and these registers by means of Load and Store instructions. The word length of the NIOS II processor is 32 bits. All registers are 32 bits long. Byte addresses in a 32-bit word can be assigned in either little-endian or big-endian style. The assignment style is one of the options that the user may select at configuration time. In this tutorial, we will use the little-endian assignment in which the lower byte addresses are used for the less significant bytes (the rightmost bytes) of the word.

The NIOS II architecture uses separate instruction and data buses, which is often referred to as the Harvard architecture.
A NIOS II processor may operate in the following three modes:

- **Supervisor mode** allows the processor to execute all instructions and perform all available functions. When the processor is reset, it enters this mode.

- **User mode** the intent of this mode is to prevent execution of some instructions that should be used for systems purposes only. Some processor features are not accessible in this mode.

- **Debug mode** is used by software debugging tools to implement features such as breakpoints and watch points.

Application programs can be run in either the User or Supervisor modes. Presently available versions of the NIOS II processor do not support the User mode.

### 2.5.2. Register Structure

The NIOS II processor has thirty two 32-bit general purpose registers, as shown in Table 2.10. Some of these registers are intended for a specific purpose and have special names that are recognized by the Assembler.

- Register \( r0 \) is referred to as the *zero* register. It always contains the constant 0. Thus, reading this register returns the value 0, while writing to it has no effect.

- Register \( rl \) is used by the Assembler as a temporary register; it should not be referenced in user programs.

- Registers \( r24 \) and \( r29 \) are used for processing of exceptions; they are not available in User mode.

- Registers \( r25 \) and \( r30 \) are used exclusively by the JTAG Debug module.

- Registers \( r27 \) and \( r28 \) are used to control the stack used by the NIOS II processor.

- Register \( r31 \) is used to hold the return address when a subroutine is called.
Table 2.10: General Purpose Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>r0</td>
<td>zero</td>
<td>0x00000000</td>
</tr>
<tr>
<td>r1</td>
<td>at</td>
<td>Assembler Temporary</td>
</tr>
<tr>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r24</td>
<td>et</td>
<td>Exception Temporary (1)</td>
</tr>
<tr>
<td>r25</td>
<td>bt</td>
<td>Breakpoint Temporary (2)</td>
</tr>
<tr>
<td>r26</td>
<td>gp</td>
<td>Global Pointer</td>
</tr>
<tr>
<td>r27</td>
<td>sp</td>
<td>Stack Pointer</td>
</tr>
<tr>
<td>r28</td>
<td>fp</td>
<td>Frame Pointer</td>
</tr>
<tr>
<td>r29</td>
<td>ea</td>
<td>Exception Return Address (1)</td>
</tr>
<tr>
<td>r30</td>
<td>ba</td>
<td>Breakpoint Return Address (2)</td>
</tr>
<tr>
<td>r31</td>
<td>ra</td>
<td>Return Address</td>
</tr>
</tbody>
</table>

(1) The register is not available in User mode  
(2) The register is used exclusively by the JTAG Debug module

There are six 32-bit control registers, as indicated in Table 2.11. The names given in the Table 2.11 are recognized by the Assembler. These registers are used automatically for control purposes. They can be read and written to by special instructions rdctl and wrctl, which can be executed only in the supervisor mode. The registers are used as follows:

- Register ctl0 reflects the operating status of the processor. Only two bits of this register are meaningful:
  - U is the User / Supervisor mode bit; U = 1 for User mode, while U = 0 for Supervisor mode.
  - PIE is the processor interrupt-enable bit. When PIE = 1, the processor may accept external interrupts. When PIE = 0, the processor ignores external interrupts.
- Register `ctl1` holds a saved copy of the status register during exception processing. The bits `EU` and `EPIE` are the saved values of the status bits `U` and `PIE`.
- Register `ctl2` holds a saved copy of the status register during debug break processing. The bits `BU` and `BPIE` are the saved values of the status bits `U` and `PIE`.
- Register `ctl3` is used to enable individual external interrupts. Each bit corresponds to one of the interrupts `irq0` to `irq31`. The value of 1 means that the interrupt is enabled, while 0 means that it is disabled.
- Register `ctl4` indicates which interrupts are pending. The value of a given bit, `ctl4k`, is set to 1 if the interrupt `irqk` is both active and enabled by having the interrupt-enable bit, `ctl3k`, set to 1.
- Register `ctl5` holds a value that uniquely identifies the processor in a multiprocessor system.

Table 2.11: Control Register

<table>
<thead>
<tr>
<th>Register</th>
<th>Name</th>
<th><code>b_{31} ... b_2</code></th>
<th><code>b_1</code></th>
<th><code>b_0</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ctl0</code></td>
<td>status</td>
<td>Reserved</td>
<td><code>U</code></td>
<td>PIE</td>
</tr>
<tr>
<td><code>ctl1</code></td>
<td>estatus</td>
<td>Reserved</td>
<td><code>EU</code></td>
<td>EPIE</td>
</tr>
<tr>
<td><code>ctl2</code></td>
<td>bstatus</td>
<td>Reserved</td>
<td><code>BU</code></td>
<td>BPIE</td>
</tr>
<tr>
<td><code>ctl3</code></td>
<td>ienable</td>
<td>Interrupt-enable</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>ctl4</code></td>
<td>ipending</td>
<td>Pending-interrupt</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>ctl5</code></td>
<td>cpuid</td>
<td>Unique processor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.5.3. Accessing Memory and I/O Devices

Figure 2.17 shows how a NIOS II processor can access memory and I/O devices. For best performance, the NIOS II/f processor can include both instruction and data caches. The caches are implemented in the FPGA memory blocks. Their usage is optional and they are specified (including their size) at the system generation time by using the SOPC Builder. The NIOS II/s version can have the instruction cache but not the data cache. The NIOS II/e version has neither instruction nor data cache.

Another way to give the processor fast access to the on-chip memory is by using the tightly coupled memory arrangement, in which case the processor accesses the memory via a direct path rather than through the Avalon network. Accesses to a tightly coupled memory bypass the cache memory. There can be one or more tightly coupled instruction and data memories. If the instruction cache is not included in a system, then there must be at least one tightly coupled memory provided for NIOS II/f and NIOS II/s processors. On-chip memory can also be accessed via the Avalon network.

Off-chip memory devices, such as SRAM, SDRAM, and Flash memory chips are accessed by instantiating the appropriate interfaces. The input/output devices are memory mapped and can be accessed as memory locations. Data accesses to memory locations and I/O interfaces are performed by means of Load and Store instructions, which cause data to be transferred between the memory and general purpose registers.
2.5.4. Addressing

The NIOS II processor issues 32-bit addresses. The memory space is byte-addressable. Instructions can read and write words (32 bits), half words (16 bits), or bytes (8 bits) of data. Reading or writing to an address that does not correspond to an existing memory or I/O location produces an undefined result.

There are five addressing modes provided:

- **Immediate mode** a 16-bit operand is given explicitly in the instruction. This value may be sign extended to produce a 32-bit operand in instructions that perform arithmetic operations.
- **Register mode** the operand is in a processor register

- **Displacement mode** the effective address of the operand is the sum of the contents of a register and a signed 16-bit displacement value given in the instruction

- **Register indirect mode** the effective address of the operand is the contents of a register specified in the instruction. This is equivalent to the displacement mode where the displacement value is equal to 0.

- **Absolute mode** a 16-bit absolute address of an operand can be specified by using the displacement mode with register r0 which always contains the value 0.
REFERENCES


