PIC32 Overview

E155
Outline

- PIC$^{32}$ Architecture
- MIPS M4K Core
- PIC$^{32}$ Peripherals
- PIC$^{32}$ Basic Operations
- Clock
Approximately $16B of microcontrollers were sold in 2011, and the market continues to grow by about 10% a year.

Microcontrollers have become ubiquitous and nearly invisible, with an estimated 150 in each home and 50 in each automobile in 2010.

The 8051 is a classic 8-bit microcontroller originally developed by Intel in 1980 and now sold by a host of manufacturers.

Microchip’s PIC16 and PIC18-series are 8-bit market leaders. **PIC (Peripheral Interface Controller)**
The Atmel AVR series of microcontrollers has been popularized among hobbyists as the brain of the Arduino platform.

Among 32-bit microcontrollers, Renesas leads the overall market, while ARM is a major player in mobile systems including the iPhone. Freescale, Samsung, Texas Instruments, and Infineon are other major microcontroller manufacturers.

Focus on the PIC32MX675F512H, a member of Microchip’s PIC32-series of microcontrollers based on the 32-bit MIPS microprocessor.
Microcontrollers

- The PIC32 family also has a generous assortment of on-chip peripherals and memory
- Selected this family
  - Inexpensive, easy-to-use development environment
  - Based on the MIPS
- Microchip is a leading microcontroller vendor that sells more than a billion chips a year.
- Microcontroller I/O systems are quite similar from one manufacturer to another, so the principles illustrated on the PIC32 can readily be adapted to other microcontrollers.
MIP Core

MIP Core

PIC$^{32}$(PIC32MX675F512H)

- MIPSM4K Core
  - Up to 80 MHz, 1.65 DMIPS/MHz
  - 5 Stage Pipeline devices
  - Instruction Trace
  - Temperature Range: -40°C to 105°C
  - AEC-Q100 qualified

- Data and Code
  - Up to 512 KB Flash (program code)
  - Up to 64 KB SRAM (data)
  - Prefetch Buffer 256KB Cache
  - Separate Buses for Instructions and Data
Connected Peripherals with DMA
- Full-speed USB Host/Device/OTG
- 10/100 Ethernet MAC with MII/RMII Interfaces
- 2x CAN 2.0B Ports
- Up to 6 UART, 4 I²C™, 3 SPI Ports, CTMU and I S
- Up to 8 Additional Channels of General Purpose DMA
## PIC\textsuperscript{32} Peripheral Features

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Family</td>
<td>PIC32MX6xx</td>
</tr>
<tr>
<td>Max Speed MHz</td>
<td>80</td>
</tr>
<tr>
<td>Program Memory Size (KB)</td>
<td>512</td>
</tr>
<tr>
<td>RAM (KB)</td>
<td>64</td>
</tr>
<tr>
<td>Temperature Range (C)</td>
<td>-40 to 105</td>
</tr>
<tr>
<td>Operating Voltage Range (V)</td>
<td>2.3 to 3.6</td>
</tr>
<tr>
<td>DMA Channels</td>
<td>8</td>
</tr>
<tr>
<td>SPI\textsuperscript{TM}</td>
<td>3</td>
</tr>
<tr>
<td>I\textsuperscript{2}C\textsuperscript{TM} Compatible</td>
<td>4</td>
</tr>
<tr>
<td>CTMU</td>
<td>NO</td>
</tr>
<tr>
<td>USB (Channels, Speed, Compliance)</td>
<td>1, FS Host/OTG, USB 2.0 OTG</td>
</tr>
<tr>
<td>A/D channels</td>
<td>16</td>
</tr>
<tr>
<td>Max A/D Sample Rate (KSPS)</td>
<td>1000</td>
</tr>
<tr>
<td>Input Capture</td>
<td>5</td>
</tr>
<tr>
<td>Output Compare/Std. PWM</td>
<td>5</td>
</tr>
<tr>
<td>16-bit Digital Timers</td>
<td>5</td>
</tr>
<tr>
<td>Parallel Port</td>
<td>PMP</td>
</tr>
<tr>
<td>Comparators</td>
<td>2</td>
</tr>
<tr>
<td>Internal Oscillator</td>
<td>8 MHz, 32 kHz</td>
</tr>
<tr>
<td>RTCC</td>
<td>Yes</td>
</tr>
<tr>
<td>I/O Pins</td>
<td>53</td>
</tr>
<tr>
<td>Pin Count</td>
<td>64</td>
</tr>
</tbody>
</table>

### PIC32 Virtual Memory Map

<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xFFFFFFF</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xBFC03000</td>
<td>Device Configuration Registers</td>
</tr>
<tr>
<td>0xBFC02FF</td>
<td>Boot Flash</td>
</tr>
<tr>
<td>0xBFC02FF0</td>
<td>SFRs</td>
</tr>
<tr>
<td>0xBFC02FEF</td>
<td>Program Flash</td>
</tr>
<tr>
<td>0xBFC00000</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xBF900000</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xBF8FFFF</td>
<td>SFRs</td>
</tr>
<tr>
<td>0xBF800000</td>
<td>Program Flash</td>
</tr>
<tr>
<td>0xBD080000</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xBD07FFFF</td>
<td>RAM</td>
</tr>
<tr>
<td>0xBD000000</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xA0020000</td>
<td>Reserved</td>
</tr>
<tr>
<td>0xA0001FFFF</td>
<td>RAM</td>
</tr>
<tr>
<td>0xA0000000</td>
<td>RAM</td>
</tr>
</tbody>
</table>
PIC32 PinOut and Package

PIC32MX6xxFxxH pinout. Black pins are 5 V-tolerant

PIC32 in 64-pin TQFP package

uMudd32 Board
PIC Basic Operation
• Program the microcontroller is with a Microchip *In Circuit Debugger* (ICD) 3, or a puck.

• Communicate with the PIC32 from a PC to download code and to debug the program.

• Connects to a USB port on the PC and to a six-pin RJ-11 modular connector (US telephone jacks) on the PIC32 development board.

• The ICD3 communicates with the PIC over a 2-wire In-Circuit Serial Programming interface with a clock and a bidirectional data pin.

• Programming by using Microchip’s free MPLAB Integrated Development Environment (IDE) to write your programs in assembly language or C, debug them in simulation, and download and test them on a development board by means of the ICD.
Timers

- 5 timers:
  - Timer1 (Type A)
  - Timers 2/3, Timers4/5 (TypeB) 16-bit timer/counter

- Common features:
  - Software-selectable internal or external clock source
  - Programmable interrupt generation and priority
  - Gated external pulse counter

Timers

- **Type A:**
  - Asynchronous timer/counter with a built-in oscillator
  - Operational during CPU Sleep mode
  - Software selectable prescalers 1:1, 1:8, 1:64 and 1:256

- **Type B:**
  - Ability to form a 32-bit timer/counter
  - Software prescalers 1:1, 1:2, 1:4, 1:8, 1:16, 1:32, 1:64 and 1:256
  - Event trigger capability

Type B Timer Block Diagram

Figure 14-1: Type A Timer Block Diagram

Note:
1. For information on enabling the 32 kHz Secondary Oscillator (SOSC), refer to Section 6, "Oscillators" (DS61112).
2. The default state of the SOSCEN bit (OSCCON<1>) during a device reset is controlled by the FSOSCEN bit (DEVCFG1<5>).

Timer Registers

- **TxCON**: 16-bit control register associated with the timer
- **TMRx**: 16-bit timer count register
- **PRx**: 16-bit register associated with the timer

## Timer1 Register

Register 14-1: T1CON: Type A Timer Control Register

<table>
<thead>
<tr>
<th>Bit 31</th>
<th>Bit 24</th>
<th>Bit 23</th>
<th>Bit 16</th>
<th>Bit 15</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>r-0</td>
<td>r-0</td>
<td>r-0</td>
<td>r-0</td>
<td>r-0</td>
<td>r-0</td>
<td>r-0</td>
<td>r-0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 31</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W-0</td>
<td>r-0</td>
<td>R/W-0</td>
<td>R/W-0</td>
</tr>
<tr>
<td>ON(1)</td>
<td>FRZ(2)</td>
<td>SIDL</td>
<td>TWDIS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TWIP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 15</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/W-0</td>
<td>r-0</td>
</tr>
<tr>
<td>TGATE</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 7</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>r-0</td>
</tr>
<tr>
<td>TCKPS&lt;1:0&gt;</td>
</tr>
<tr>
<td>—</td>
</tr>
<tr>
<td>TSYNC</td>
</tr>
<tr>
<td>TCS</td>
</tr>
<tr>
<td>—</td>
</tr>
<tr>
<td>bit 0</td>
</tr>
</tbody>
</table>

Legend:
- **R** = Readable bit
- **W** = Writable bit
- **P** = Programmable bit
- **r** = Reserved bit
- **U** = Unimplemented bit
- **-n** = Bit Value at POR: (0, 1, x = Unknown)

Timer1 Control Registers

- bit 31-16 **Reserved**: Write ‘0’; ignore read
- bit 15 **ON**: Timer On bit\(^{(1)}\)
  - 1 = Timer is enabled
  - 0 = Timer is disabled
- bit 14 **FRZ**: Freeze in Debug Exception Mode bit\(^{(2)}\)
  - 1 = Freeze operation when CPU is in Debug Exception mode
  - 0 = Continue operation even when CPU is in Debug Exception mode
  - bit 13 **SIDL**: Stop in Idle Mode bit
    - 1 = Discontinue operation when device enters Idle mode
    - 0 = Continue operation even in Idle mode
- bit 12 **TWDIS**: Asynchronous Timer Write Disable bit
  - 1 = Writes to TMR1 are ignored until pending write operation completes
  - 0 = Back-to-back writes are enabled (Legacy Asynchronous Timer functionality)
- bit 11 **TWIP**: Asynchronous Timer Write in Progress bit
  - In Asynchronous Timer mode:
    - 1 = Asynchronous write to TMR1 register in progress
    - 0 = Asynchronous write to TMR1 register complete
  - In Synchronous Timer mode:
    - This bit is read as ‘0’.
  - bit 10-8 **Reserved**: Write ‘0’; ignore read

**Note 1**: When using 1:1 PBCLK divisor, the user’s software should not read/write the peripheral SFRs in the SYSCLK cycle immediately following the instruction that clears the module’s ON bit.

**Note 2**: This bit is writable only in Debug Exception mode. It is forced to ‘0’ in normal mode.
Timer 1 Control Registers

- **bit 7 TGATE**: Timer Gated Time Accumulation Enable bit
  - When TCS = 1:
  - This bit is ignored and is read as ‘0’.
  - When TCS = 0:
  - 1 = Gated time accumulation is enabled
  - 0 = Gated time accumulation is disabled

- **bit 6 Reserved**: Write ‘0’; ignore read

- **bit 5-4 TCKPS<1:0>**: Timer Input Clock Prescale Select bits
  - 11 = 1:256 prescale value
  - 10 = 1:64 prescale value
  - 01 = 1:8 prescale value
  - 00 = 1:1 prescale value

- **bit 3 Reserved**: Write ‘0’; ignore read

- **bit 2 TSYNC**: Timer External Clock Input Synchronization Selection bit
  - When TCS = 1:
  - 1 = External clock input is synchronized
  - 0 = External clock input is not synchronized
  - When TCS = 0:
  - This bit is ignored and is read as ‘0’.

- **bit 1 TCS**: Timer Clock Source Select bit
  - 1 = External clock from TxCKI pin
  - 0 = Internal peripheral clock

- **bit 0 Reserved**: Write ‘0’; ignore read

**Note 1**: When using 1:1 PBCLK divisor, the user’s software should not read/write the peripheral SFRs in the SYSCLK cycle immediately following the instruction that clears the module’s ON bit.

**2**: This bit is writable only in Debug Exception mode. It is forced to ‘0’ in normal mode.
Count to 1 sec

- TMRI prescale
  - Run periphery clock at \( \frac{1}{4} \) speed or 10MHz
  - \( P_{clk} = \frac{clk}{4} \)
  - Prescale by 256
  - Each count \((0.1\, \text{us}) \times 256 = 25.6\, \text{us}\)
  - \[ \text{Count} = \frac{1\, \text{sec}}{25.6\, \text{us}} = 39062 \]
Count to 1 sec

- Start TMR1 at 0
- Set T1CON
- Wait for TMRI=39062

```
la      $t1, TMR1
sw      $zero, 0($t1)    # reset timer1
#T1CON:
    #    bit 15 = 1    On
    #     14 = 1    FRZ  (freeze on Debug exception)
    #    5:4 = 11  Prescale by 256
ori     $t1, $0, 16'b1100_0000_0011_0000
la  $t0, T1CON
sw  $t1, 0($t0)
addi  $t4, $0, 39062
la  $t0, TMR1
poll:
    lw  $t3, 0($t0)  # t3=TMRI
bne  $t3, $t4, poll
nop
```