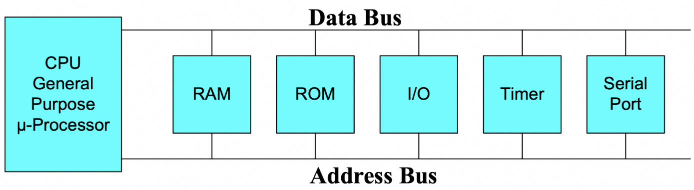
VE373 Recitation Class

Week 2

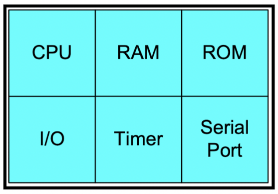
2022.05.21

L1 — Introduction to Embedded Systems

1. **Microprocessor (MPU) & Microcontroller (MCU)**



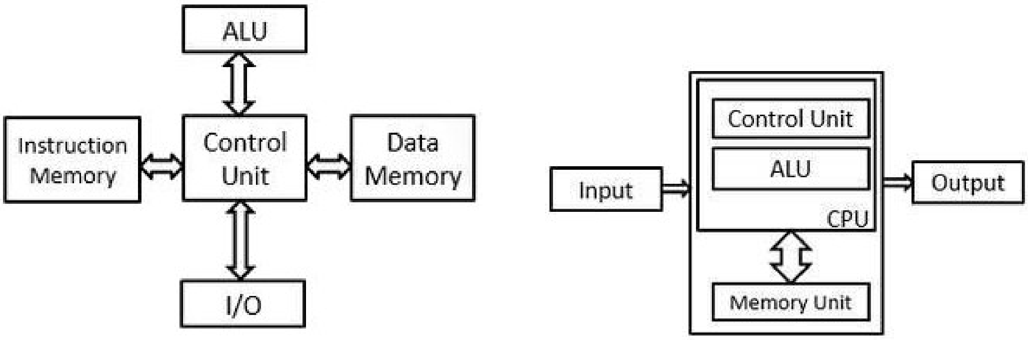
(a) Microprocessor



(b) Microcontroller

Microprocessor	Micro Controller
Microprocessor is heart of Computer system.	Micro controller is a hear of embedded system.
It is just a processor. Memory and I/O components have to be connected externally.	**Micro controller has external processor along with internal memory and I/O components.**
The circuit is large.	The circuit is small.
Cannot be used in compact systems and hence inefficient.	Can be used in compact systems and hence it is an efficient technique.
Cost of the entire system increases.	Cost of the entire system is low.
Due to external components, the entire power consumption is high. Hence it is not suitable to used with devices running on stored power like batteries.	Since external components are low, total power consumption is less and can be used with devices running on stored power like batteries.
Most of the microprocessors do not have power saving features.	Most of the micro controllers have power saving modes like idle mode and power saving mode. This helps to reduce power consumption even further.
Since memory and I/O components are all external, each instruction will need external operation, hence it is relatively slower.	Since components are internal, most of the operations are internal instruction, hence speed is fast.
Microprocessor have less number of registers, hence more operations are memory based.	Micro controller have more number of registers, hence the programs are easier to write
Microprocessors are based on Von Neumann model/architecture where program and data are stored in same memory module.	Micro controllers are based on Harvard architecture where program memory and Data memory are separate.
Mainly used in personal computers.	Used mainly in washing machines, MP2 players.

2. **Von Neumann & Havard Architecture**



(a) Harvard Architecture (b) Von Neumann Architecture

Harvard architecture	Von Neumann architecture
It required two memories for their instruction and data.	**It required only one memory for their instruction and data.**
Harvard architecture is required separate bus for instruction and data.	Von Neumann architecture is required only one bus for instruction and data.
Processor can complete an instruction in one cycle	Processor needs two clock cycles to complete an instruction.
Easier to pipeline, so high performance can be achieve.	Low performance as compared to Harvard architecture.
Comparatively high cost.	It is cheaper.

L2 — PIC MCU Architecture

1. **PIC32**

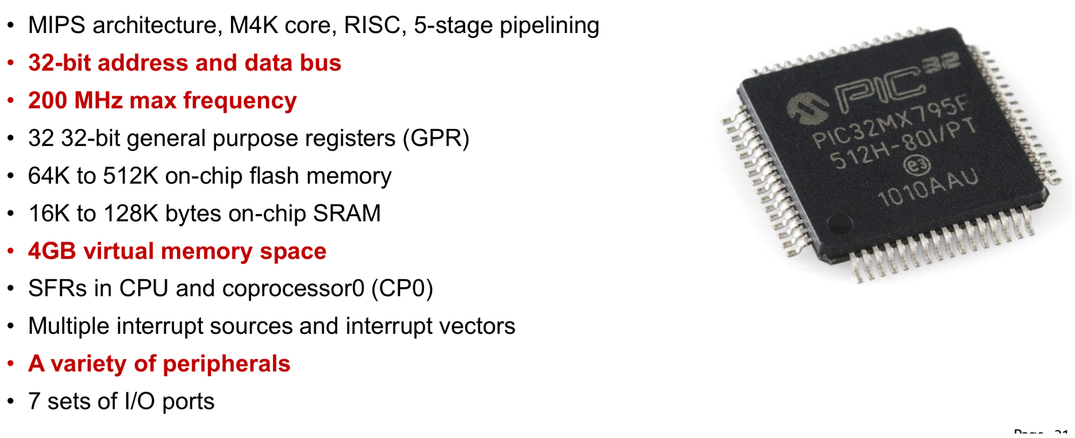


Figure 3: 32-bit PIC32 MCU Example: Architecture

n-bit data address bus = support up to 2^n byte memory.

PIC32 is a combination of Von Neumann and Hasrvard architecture. See here for more details if interested.

2. **Special Function Register**

Used to config, control and monitor various aspects of the microprocessor's function.

For example, `T1CON`, `TMR1` and `PR1` when using Timer 1.

L3 — Embedded Programming

1. **Compiler**

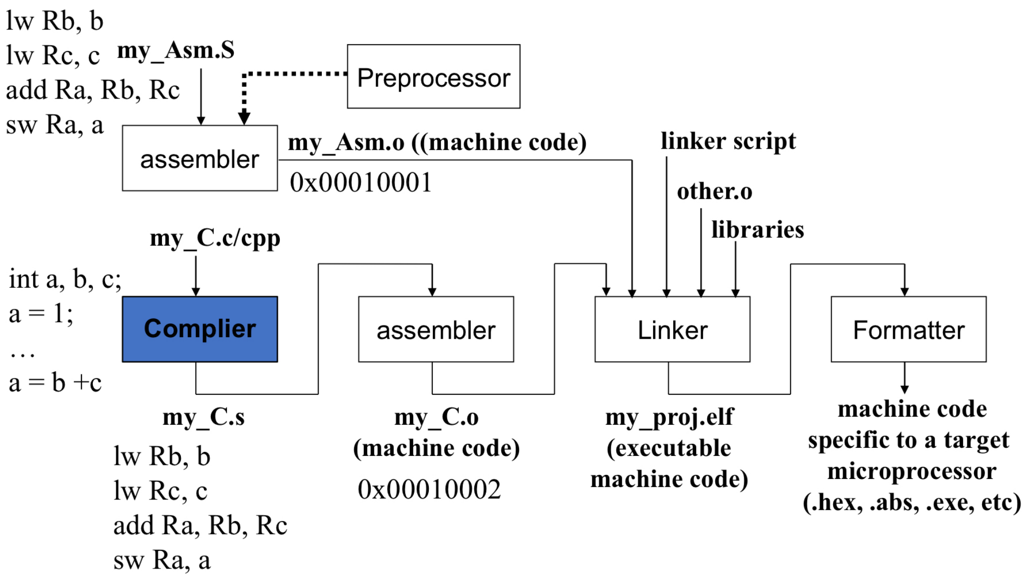


Figure 4: Workflow

2. **Const & Volatile**
   - const: the value not supposed to be written by program (read only).
   - volatile: the value can be changed by something other than program so it should be reexamined frequently.

3. **Directives**

   Directives: compiler dependent commands.

   e.g. `#pragma interrupt func_name ipln` (will be used later).

L4 — Timers and IO

1. **Oscillator**
   - **Internal oscillator**:
     - Integrated on-chip within the processor
     - Low frequency accuracy and stability.
     - Most of internal oscillator are RC circuits
   - **External oscillator**
     - Located off-chip on the PCB
     - High frequency accuracy and stability
     - Most of external oscillator are crystal oscillator

   Peripherals don't have a high clock frequency:

   - No need
   - Limited by parasitics
   - Power consumption

2. **Timer**

   Two types, five timers on PIC32:

- Type A — Timer 1
  - 16-bit synchronous/asynchronous timer/event counter
  - Internal or external clock source
  - Gated external event timer
- Type B — Timer 2, 3, 4, 5
  - 16/32-bit synchronous timer/event counter
  - Internal or external clock source
  - Gated external event timer

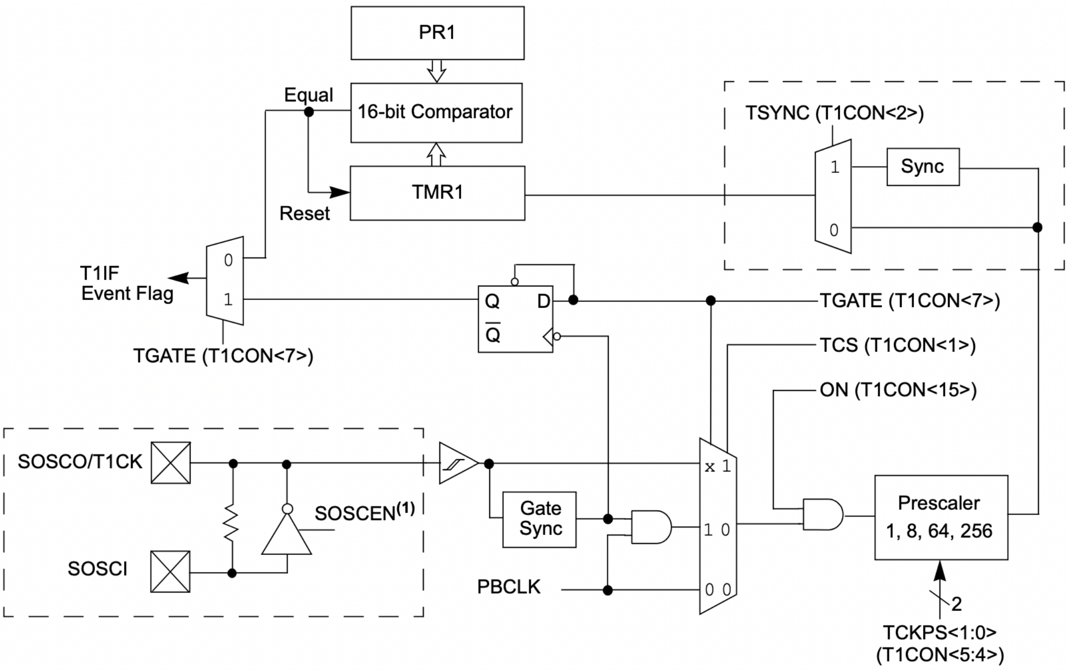


Figure 5: Timer 1 block diagram (Datasheet Page. 197)

3. **16-bit synchronous timer setup steps**

   Use internal clock source — `PBCLK` as example.

   1. Clear control bit `ON` (`TxCON<15>=0`) to disable timer.
   2. Clear control bit `TCS` (`TxCON<1>=0`) to select internal clock source `PBCLK`.
   3. Select desired clock prescale value.
   4. Load/clear timer register `TMRx` to specify initial counting value.
   5. Load period register `PRx` with desired 16-bit match value.
   6. If interrupts used:
      i. Clear interrupt flag bit
      ii. Configure interrupt priority and subpriority levels
      iii. Set interrupt enable bit
   7. Set control bit `ON` (`TxCON<15>=1`) to start timer