Computer Organization and Embedded System

Hardware-Software design issues on embedded system (ACtE0404)

Ishwar Kumar Singh
Computer Engineer
Government of Nepal

4.4 Hardware-Software design issues on embedded system (ACtE0404

- Embedded Systems overview,
- Classification of Embedded Systems
- Custom Single-Purpose Processor Design,
- Optimizing Custom Single-Purpose Processors,
- Basic Architecture of Software Design and Operation,
- Programmer’s View,
- Development Environment,
- Application-Specific Instruction-Set Processors.

Embedded Systems

- An embedded system is a microprocessor-based computer hardware system with software that is designed to perform a dedicated function, either as an independent system or as a part of a large system.

Architecture of embedded systems

- Embedded system mainly has two parts: embedded hardware and embedded software.
- Embedded hardware are based around microprocessors and microcontrollers, include memory, bus, Input/Output, Controller.
- Embedded software includes embedded operating systems, different applications and device drivers.
- Architecture of the Embedded System includes Sensor, Analog to Digital Converter, Memory, Processor, Digital to Analog Converter, and Actuators etc.

Architecture of embedded systems

The basic components works as following:

- **Sensor:** The sensor measures and converts the physical quantity to an electrical signal. Also stores the measured quantity to the memory.
- **A-D Converter:** An analog-to-digital converter converts the analog signal sent by the sensor into a digital signal.
- **Processor:** Processors assess the data to measure the output and store it to the memory.
- **D-A Converter:** A digital-to-analog converter changes the digital data fed by the processor to analog data
- **Actuator:** An actuator compares the output given by the D-A Converter to the actual (expected) output stored in it and stores the approved output.

Classification of Embedded Systems

Embedded Systems are classified based on the two factors:

- Performance and Functional Requirements
- Performance of Micro-controllers

Performance & Functional Requirements

Embedded Systems are divided into 4 types as follows:

- Real-Time Embedded Systems
- Stand Alone Embedded Systems
- Networked Embedded Systems
- Mobile Embedded Systems

Performance of Micro-controllers

Embedded Systems are divided into 3 types as follows:

- Small Scale Embedded Systems
- Medium Scale Embedded Systems
- Sophisticated or Complex Embedded Systems

Real-Time Embedded Systems

- A Real-Time Embedded System is strictly time specific which means these embedded systems provides output in a particular/defined time interval.
- These type of embedded systems provide quick response in critical situations which gives most priority to time based task performance and generation of output.
- Real time embedded systems are used in defence sector, medical and health care sector, and some other industrial applications where output in the right time is given more importance.
- **Examples:**
  - Traffic control system
  - Military usage in defences sector
  - Medical usage in health sector

Stand Alone Embedded Systems

- Stand Alone Embedded Systems are independent systems which can work by themselves they don’t depend on a host system.
- It takes input in digital or analog form and provides the output.
- Examples:
  - MP3 players
  - Microwave ovens
  - Calculator

Networked Embedded Systems

- Networked Embedded Systems are connected to a network which may be wired or wireless to provide output to the attached device.
- They communicate with embedded web server through network.
- Examples:
  - Home security systems
  - ATM machine
  - Card swipe machine

Mobile Embedded Systems

- Mobile embedded systems are small and easy to use and requires less resources.
- They are the most preferred embedded systems due its portability.
- Examples:
  - Mobile Phones
  - Digital Camera

Classification of Embedded Systems

Embedded Systems are classified based on the two factors:

- Performance and Functional Requirements
- Performance of Micro-controllers

Performance & Functional Requirements

Embedded Systems are divided into 4 types as follows:

- Real-Time Embedded Systems
- Stand Alone Embedded Systems
- Networked Embedded Systems
- Mobile Embedded Systems

Performance of Micro-controllers

Embedded Systems are divided into 3 types as follows:

- Small Scale Embedded Systems
- Medium Scale Embedded Systems
- Sophisticated or Complex Embedded Systems

Small Scale Embedded Systems

- Small Scale Embedded Systems are designed using an 8-bit or 16-bit micro-controller.
- They can be powered by a battery.
- The processor uses very less/limited resources of memory and processing speed.
- Mainly these systems does not act as an independent system they act as any component of computer system.

(do not compute and dedicated for a specific task)

Medium Scale Embedded Systems

- Medium Scale Embedded Systems are designed using an 16-bit or 32-bit micro-controller.
- These medium Scale Embedded Systems are faster than that of small Scale Embedded Systems.
- Integration of hardware and software is complex in these systems.
- Java, C, C++ are the programming languages are used to develop medium scale embedded systems.
- Different type of software tools like compiler, debugger, simulator, etc. are used to develop these type of systems.

Complex Embedded Systems

- Also known as Sophisticated Embedded Systems
- Complex Embedded Systems are designed using multiple 32-bit or 64-bit micro-controller.
- These systems are developed to perform large scale complex functions.
- These systems have high hardware and software complexities.

Custom Single Purpose Processor Design

- Processor is a digital circuit that performs computational tasks.
- The minimum requirement to be a processor is the presence of **controller** and **data path**.
- The different processor technologies are as follows:
  - General Purpose Processor (GPP)
  - Single Purpose Processor (SPP)
  - Application Specific Processor (ASP

Advantages

1. Faster performance
2. Size is smaller
3. Lower power management
4. High NRE cost
5. Longer Time to market
6. Less Flexible

Processor Design

General Purpose Processor (GPP)

- It is a programmable circuit that can perform **varieties of tasks**.
- It consists of **program memory** and **general data path**.
- The data path has **large register array** and one or more general purpose ALU.

Single Purpose Processor (SPP)

- It is a digital circuit designed to perform **exactly one program**.
- It only consists of **data memory** but not program memory (Ex: Digital Camera).

Processor Design

Application Specific Processor (ASP)

- It is a programmable circuit that is optimized for a **particular class of applications**.
- It consists of program memory, data memory, custom designed ALU and optimized datapath.

Processor Design

Designing custom single purpose processor

1. Develop an **algorithm or function** that computes the desired output.

2. Convert algorithm into a **complex state diagram** or **finite state machine with data (FSMD).**

3. Divide **functionality** with **datapath port** and **controller part**.

   - Datapath consists of **interconnection** of **combinational** and **sequential** components.
   - Controller consists of pure **finite state machine (FSM).**

4. Complete **controller design** by implementing **finite state machine** with combinational logic.

Optimizing Custom Single Purpose Processor

- Optimization is the technique of improving the design metrics so as to get the best possible values of various design metrics.
- The optimization opportunities in Custom Single Purpose Processor are as follows:-

1. Optimization of original program:
2. Optimizing FSMD
3. Optimizing Datapath
4. Optimizing FSM

Optimization

Optimization of original program:

- The algorithms are analysed in terms of **time complexity** and **space complexity** and focus of developing more **efficient** alternative algorithms.
- It involves decreasing of number of computations and size of variables if possible.

Optimizing FSMD

- Focus on optimizing the **data processing paths**
- Reducing the **number of operations** of simplifying them
- Simplifying the **control logic** that manages **state transitions** and **data operations**

Optimization

Optimizing Datapath

- Many functional operations can share a single functional unit if those operations occur in different stages.
- Subtractor can be used and selection can be done using multiplexor.

Optimizing FSM

- FSM can be optimized using **state encoding** and **state minimization**.
- State encoding is the task of **assigning** a unique bits to **encode** ‘n’ states to reduce the **complexity** of the circuit or software implementation.
- State minimization is the task of merging equivalent states into a single state.

Basic Architecture of Software Design and Operation

- It consists of general datapath.
- Control unit does not store algorithms
- Algorithm is programmed into memory.

Operations

- Fetch Instruction: It gets the next instructions as indicated by location in memory pointed by PC and stores into IR.
- Decode Instruction: It determines the actual operations performed by the instructions at IR.
- Fetch Operands: It gets the operand data needed for instruction from memory to appropriate registers of datapath.

Basic Architecture of Software Design and Operation

- Execute: The actual arithmetic or logical operations is performed by moving data through ALU.
- State Results: It writes the data from datapath register into memory .

Pipelining

- Pipelining is the mechanism to increase instruction throughput of a microprocessor.
- It assumes the independent operations to be performed simultaneously.

Programmer's View

- Programmer does not need detailed understanding of architecture.
- They just need to understand which instructions can be executed.

Programmer's Consideration

1. Program and Data memory space
2. Registers
3. Input Output (I/O)
4. Interrupts
5. Operating System

Development Environment

- **Development Processor:** the processor on which programs are written.
- **Target Processor:** The processor that will run the program.

- If Development and Target processor are different, the code can be run by downloading to target processor or by simulation.
- Simulation can be done by using HDL and Instruction set simulator(ISS)

Application Specific Instruction Set Processor

- ASIP is the processor targeted for a particular domain i.e. domain specific but can be programmed. For example Microcontroller, Digital Signal Processor.
- It is a type of microprocessor specifically designed to efficiently execute a particular application domain.
- **ASIPs are optimized for a specific set of applications, resulting in improved performance, power efficiency, and cost-effectiveness for these targeted tasks.**
- ASIPs are built by tailoring the **instruction set architecture** (ISA) to match the requirements of the target application domain.
- This customization allows for the inclusion of specialized instructions and hardware accelerators that can significantly enhance the execution of specific algorithms or tasks.

Thank You.