Embedded Processors in SoC Design
Embedded microprocessors

ARM7TDMI
Example: Connected Mobile Computer
ARM7TDMI Core Signals

Clocks
- MCLK
- nWAIT

Configuration
- ISYNC
- BUSSEN
- BIGEND

Interrupts
- nIRQ
- nFIQ
- nRESET

Bus Control
- APE
- DBE
- VDD
- VSS

Power

Memory Interface
- nM[4:0]
- nMREQ
- SEQ
- LOCK
- A[31:0]
- D[31:0]
- DIN[31:0]
- DOUT[31:0]
- nENOUT
- nRW
- MAS[1:0]
- nTRANS
- ABORT
- nOPC
- nCPI
- CPA
- CPB

Memory Management

Coprocessor Interface
ARM7TDMI Block Diagram

ARM7TDMI Core

TAP Controller

JTAG Interface

Control Signals

Address Bus

Data Bus

Bus Splitter

DIN[31:0]

DOUT[31:0]

A[31:0]

D[31:0]

Embedded ICE Logic
The ARM7TDM Core
Data processing Instructions

- Consist of:
  - Arithmetic: ADD ADC SUB SBC RSB RSC
  - Logical: AND ORR EOR BIC
  - Comparisons: CMP CMN TST TEQ
  - Data movement: MOV MVN

- These instructions only work on registers, NOT memory.

- Syntax:

  `<Operation>{<cond>}{S} Rd, Rn, Operand2`

  - Comparisons set flags only - they do not specify Rd
  - Data movement does not specify Rn

- Second operand is sent to the ALU via barrel shifter.
Using the Barrel Shifter

Register, optionally with shift operation
- Shift value can be either be:
  - 5 bit unsigned integer
  - Specified in bottom byte of another register.
- Used for multiplication by constant

Immediate value
- 8 bit number, with a range of 0-255.
  - Rotated right through even number of positions
- Allows increased range of 32-bit constants to be loaded directly into registers
Single register data transfer

- **LDR**   **STR**  Word
- **LDRB**  **STRB**  Byte
- **LDRH**  **STRH**  Halfword
- **LDRSB** Signed byte load
- **LDRSH** Signed halfword load

- Memory system must support all access sizes

- Syntax:
  - \texttt{LDR\{<cond>\}{<size>} Rd, <address>}
  - \texttt{STR\{<cond>\}{<size>} Rd, <address>}

  e.g. \texttt{LDREQB}
Branch instructions

- Branch: \( B\{\langle\text{cond}\rangle\} \text{ label} \)
- Branch with Link: \( BL\{\langle\text{cond}\rangle\} \text{ subroutine\_label} \)

The processor core shifts the offset field left by 2 positions, sign-extends it and adds it to the PC
- ± 32 Mbyte range
- How to perform longer branches?
The Instruction Pipeline

- The ARM7 family uses a 3 stage pipeline in order to increase the speed of the flow of instructions to the processor.
  - Allows several operations to be undertaken simultaneously, rather than serially.

<table>
<thead>
<tr>
<th>ARM</th>
<th>Thumb</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>PC</td>
</tr>
<tr>
<td>PC - 4</td>
<td>PC - 2</td>
</tr>
<tr>
<td>PC - 8</td>
<td>PC - 4</td>
</tr>
</tbody>
</table>

- Instruction fetched from memory
- Decoding of registers used in instruction
- Register(s) read from Register Bank
  - Shift and ALU operation
  - Write register(s) back to Register Bank

- The PC points to the instruction being fetched, not the instruction being executed.
Optimal Pipelining

In this example it takes 6 clock cycles to execute 6 instructions.
All operations here are on registers (single cycle execution).
Clock cycles per Instruction (CPI) = 1
# LDR Pipeline Example

In this example it takes 6 clock cycles to execute 4 instructions

- Clock cycles per Instruction (CPI) = 1.5
Branch Pipeline Example

- Breaking the pipeline
- Note that the core is executing in ARM state
Pipeline changes for ARM9TDMI

ARM7TDMI

FETCH

Instruction Fetch

DECODE

Thumb→ARM decompress
ARM decode
Reg Select

EXECUTE

Reg Read
Shift
ALU
Reg Write

ARM9TDMI

FETCH

Instruction Fetch

DECODE

ARM or Thumb Inst Decode
Reg Decode
Reg Read

EXECUTE

Shift + ALU

MEMORY

Memory Access

WRITE

Reg Write
Longer Pipelining

In this example it takes 6 cycles to execute 6 instructions, CPI of 1.

The LDR instruction does not cause the pipeline to interlock.
ARM10 vs. ARM11 Pipelines

**ARM10**

- Instruction Fetch
- ARM or Thumb Instruction Decode
- Reg Read
- Shift + ALU
- Memory Access
- Multiply
- Multiply Add
- Reg Write

**ARM11**

- Fetch
- Decode
- Issue
- MAC 1
- MAC 2
- MAC 3
- Write back
- Address
- Data Cache 1
- Data Cache 2
An Example AMBA System

High Performance ARM processor

High-bandwidth on-chip RAM

High Performance
Pipelined
Burst Support
Multiple Bus Masters

APB Bridge

AHB

 UART
Timer
Keypad
PIO

Low Power
Non-pipelined
Simple Interface

High Bandwidth
External Memory Interface

High Bandwidth
External Memory Interface
AHB Structure
High-End Car of 2012

- GPS
- Biometrics
- 100 GB HD
- Cell Phone
- Bluetooth
- NFC / RFID
- Gas Station/Garage
- Digital Content – Audio/Local Info
- Car Servicing Data
- Fuel Cell
- Fault Tolerant Network
- Chassis Control
- Drive-by-Wire

Autosar Enabled
Concept Smart Phone of 2008

- 1 billion transistor chips
- ARM11™ MPcore™ 1GHz CPU
- VGA H.264 encode
- GPU (PSP level performance)
- 256MB DRAM, 64MB Flash
- WVGA screen (852x480)
- 4 mega pixel camera

- GPS
- Stereo Headset
- Bluetooth/UWB
- NFC / RFID
- Biometrics
- 20 GB HD
- 3.5G (HSDPA) WiMax (10 Mbit down)
- DMB (Digital Mobile Broadcast)
- TV out
- PC / Mac
- 512 MB Memory card
- NFC / RFID
- Stereo Headset
- Bluetooth/UWB
- NFC / RFID
- TV out
- PC / Mac
- 512 MB Memory card
Questions or comments:
university@arm.com