World population hits 7 billion on Oct. 31, or thereabouts

Though it’s impossible to say exactly when it will happen, demographers have chosen the date to mark the milestone. Humanity remains on a steep growth curve.

By Kenneth R. Weiss, Los Angeles Times
October 31, 2011

Population grows when births exceed deaths. The 7-billion mark was reached because people are living longer and the number of infant deaths has dropped, because of a more secure food supply and because of advances in sanitation and medicine.

Review

• CPU design involves Datapath, Control
  – 5 Stages for MIPS Instructions
    1. Instruction Fetch
    2. Instruction Decode & Register Read
    3. ALU (Execute)
    4. Memory
    5. Register Write
• Datapath timing: single long clock cycle or one short clock cycle per stage

Datapath and Control

• Datapath based on data transfers required to perform instructions
• Controller causes the right transfers to happen

CPU Clocking (1/2)

• For each instruction, how do we control the flow of information though the datapath?
• Single Cycle CPU: All stages of an instruction completed within one long clock cycle
  – Clock cycle sufficiently long to allow each instruction to complete all stages without interruption within one cycle

CPU Clocking (2/2)

• Alternative multiple-cycle CPU: only one stage of instruction per clock cycle
  – Clock is made as long as the slowest stage

  1. Instruction Fetch
  2. Decode/ Register Read
  3. Execute/ Memory
  4. Register Write

  – Several significant advantages over single cycle execution: Unused stages in a particular instruction can be skipped OR instructions can be pipelined (overlapped)
Processor Design: 5 steps
Step 1: Analyze instruction set to determine datapath requirements
  – Meaning of each instruction is given by register transfers
  – Datapath must include storage element for ISA registers
  – Datapath must support each register transfer
Step 2: Select set of datapath components & establish clock methodology
Step 3: Assemble datapath components that meet the requirements
Step 4: Analyze implementation of each instruction to determine setting of control points that realizes the register transfer
Step 5: Assemble the control logic

The MIPS Instruction Formats
• All MIPS instructions are 32 bits long. 3 formats:
  – R-type
  – I-type
  – J-type
• The different fields are:
  – op: operation ("opcode") of the instruction
  – rs, rt, rd: source and destination register specifiers
  – shamt: shift amount
  – funct: selects the variant of the operation in the "op" field

The MIPS-lite Subset
• ADDU and SUBU:
  – addu rd, rs, rt
  – subu rd, rs, rt
• OR Immediate:
  – ori rt, rs, imm16
• LOAD and STORE Word:
  – lw rt, rs, imm16
  – sw rt, rs, imm16
• BRANCH:
  – beq rs, rt, imm16

Register Transfer Language (RTL)
RTL gives the meaning of the instructions
All start by fetching the instruction
\{ \text{op, } rs, rt, rd, shamt, funct \} \rightarrow \text{MEM[PC]}
\{ \text{op, } rs, rt, imm16 \} \rightarrow \text{MEM[PC]}

Instr. Register Transfers
ADDU \text{R[rd]} ← \text{R[rs]} + \text{R[rt]}; PC ← PC + 4
SUBU \text{R[rd]} ← \text{R[rs]} - \text{R[rt]}; PC ← PC + 4
ORI \text{R[rt]} ← \text{R[rs]} | \text{zero_ext[imm16]}; PC ← PC + 4
LOAD \text{R[rt]} ← \text{MEM[ R[rs] + sign_ext[imm16] ]}; PC ← PC + 4
STORE \text{MEM[ R[rs] + sign_ext[imm16] ]} ← \text{R[rt]}; PC ← PC + 4
BEQ if ( \text{R[rs]} = \text{R[rt]})
\text{then PC ← PC + 4 + (sign_ext[imm16] || 00) }$
\text{else PC ← PC + 4}
Step 1: Requirements of the Instruction Set

- Memory (MEM)
  - Instructions & data (will use one for each)
- Registers (R: 32 x 32)
  - Read RS
  - Read RT
  - Write RT or RD
- PC
- Extender (sign/zero extend)
- Add/Sub/OR unit for operation on register(s) or extended immediate
- Add 4 (+ maybe extended immediate) to PC
- Compare registers?

Step 2: Components of the Datapath

- Combinational Elements
- Storage Elements + Clocking Methodology
- Building Blocks

ALU Needs for MIPS-lite + Rest of MIPS

- Addition, subtraction, logical OR, ==:
  - ADDU R[rd] = R[rs] + R[rt]; ...
  - SUBU R[rd] = R[rs] - R[rt]; ...
  - ORI R[rt] = R[rs] | zero_ext(Imm16)...
  - BEQ if ( R[rs] == R[rt] )...
- Test to see if output == 0 for any ALU operation gives == test. How?
- P&H also adds AND, Set Less Than (1 if A < B, 0 otherwise)
- ALU follows Chapter 5

Storage Element: Idealized Memory

- Memory (idealized)
  - One input bus: Data In
  - One output bus: Data Out
- Memory word is found by:
  - Address selects the word to put on Data Out
  - Write Enable = 1: address selects the memory word to be written via the Data In bus
- Clock input (CLK)
  - CLK input is a factor ONLY during write operation
  - During read operation, behaves as a combinatorial logic block: Address valid ⇒ Data Out valid after "access time"

Storage Element: Register File

- Register File consists of 32 registers:
  - Two 32-bit output busses: busA and busB
  - One 32-bit input bus: busW
- Register is selected by:
  - RA (number) selects the register to put on busA (data)
  - RB (number) selects the register to put on busB (data)
  - RW (number) selects the register to be written via busW (data) when Write Enable is 1
- Clock input (clk)
  - CLK input is a factor ONLY during write operation
  - During read operation, behaves as a combinational logic block:
    - RA or RB valid ⇒ busA or busB valid after "access time."
Step 3a: Instruction Fetch Unit

- Register Transfer Requirements ⇒ Datapath Assembly
- Instruction Fetch
- Read Operands and Execute Operation
- Common RTL operations
  - Fetch the Instruction: mem(PC)
  - Update the program counter:
    - Sequential Code: PC ← PC + 4
    - Branch and jump: PC ← “something else”

Clocking Methodology

- Storage elements clocked by same edge
- Flip-flops (FFs) and combinational logic have some delays
  - Gates: delay from input change to output change
  - Signals at FF 0 input must be stable before active clock edge to allow
    signal to travel within the FF (set-up time), and we have the usual
    clock-to-Q delay
- “Critical path” longest path through logic determines length of clock period

Step 3b: Add & Subtract

- R[rd] = R[rs] op R[rt] (addu rd,rs,rt)
  - Ra, Rb, and Rw come from instruction’s Rs, Rt, and Rd fields
  - ALUctr and RegWr: control logic after decoding the instruction
  - ALUctr and RegWr

Register-Register Timing: One Complete Cycle

- Memory
  - Instruction Memory Access Time
  - Delay from instruction word to ALU
  - ALU Delay
- Register File
  - Register File Access Time
  - New Value

Putting it All Together: A Single Cycle Datapath

Processor Design: 3 of 5 steps

Step 1: Analyze instruction set to determine datapath requirements
- Meaning of each instruction is given by register transfers
- Datapath must include storage element for ISA registers
- Datapath must support each register transfer

Step 2: Select set of datapath components & establish clock methodology

Step 3: Assemble datapath components that meet the requirements
- Select appropriate control signals for each instruction

Step 4: Analyze implementation of each instruction to determine setting of control points that realizes
  the register transfer

Step 5: Assemble the control logic