Reconstruction of the MOS 6502 on the Cyclone II FPGA

Yu Chen (yc2615)
Jaebin Choi (jc3797)
Arthy Sundaram (as4304)
Anthony Erlinger (afe2104)
Outline

• Redesign 6502
  ➢ RTL Level structure

• 6502 opcodes summary

• 6502 on the DE2 board
  ➢ Monitoring the internal registers
  ➢ Bouncing Ball
- **Original Block Diagram**

- **Challenges:**
  - Latch based
  - Two phases clk
  - High Impedance Bus
Redesign 6502

- **Latch ➔ DFF**

  - **Single Clock**
  - **Datapath ➔ Mux**
RTL level structure

- **Control Path**
  - Predecoeder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic

- **Datapath**
  - Mux
  - Register File
  - ALU
RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic

- Datapath
  - Mux
  - Register File
RTL level structure

- Control Path
  - Predecoer
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic
- Datapath
  - Mux
  - Register File
RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic

- Datapath
  - Mux
  - Register File
Timing Generator—Mealy Machine

- Three types
  - Branch
  - RMW
  - Normal
RTL level structure

- **Control Path**
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic

- **Datapath**
  - Mux
  - Register File
6502 on the DE2 board.
6502 Instruction Set Architecture

• Each instruction is encoded within 8 bits (62 total). Which uniquely defines the

  1. **Operation** to be performed (known as the ‘**opcode**’).

  2. **Addressing mode** which defines how that operation handles and modifies memory.
Addressing modes

- 12 types of Addressing modes, but general addressing modes are:

<table>
<thead>
<tr>
<th>Opcode</th>
<th>Addressing Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>(zero page,X)</td>
</tr>
<tr>
<td>001</td>
<td>zero page</td>
</tr>
<tr>
<td>010</td>
<td>#immediate</td>
</tr>
<tr>
<td>011</td>
<td>absolute</td>
</tr>
<tr>
<td>100</td>
<td>(zero page),Y</td>
</tr>
<tr>
<td>101</td>
<td>zero page,X</td>
</tr>
<tr>
<td>110</td>
<td>absolute,Y</td>
</tr>
<tr>
<td>111</td>
<td>absolute,X</td>
</tr>
</tbody>
</table>

- Each opcode can have multiple addressing modes. Taking this into account, there are 152 possible instructions.
# General Instruction Format

Most Codes follow: **AAAABBBCCC**

**BBB**: Defines the Addressing Mode

**AAA---CC**: Defines the Opcode
Operands

Most operations (except for implied instructions such as CLC, DEX, INY, etc...) accept up to three operands which occupy the following adjacent bits in memory.

<table>
<thead>
<tr>
<th>OPCODE (8-bit)</th>
<th>OPERAND 1 (8-bit)</th>
<th>OPERAND 2 (8-bit)</th>
<th>OPERAND 3 (8-bit)</th>
</tr>
</thead>
</table>

Access to these operands and to internal and external memory is defined by a Mealy State Machine.
Debugging

• 1. Software debug: Modelsim
  - Timed databus behavior
    Ex) Databus<=x”A9”; wait for 40ns;
      Databus<=x”07”; wait for 40ns;
  - Tested ~50% of opcodes, one by one
  - Does not test data fetch, or memory structure
  - Quick debug.
Debugging

• 2. Hardware debug
  ➢ Real time
  ➢ Tests full 6502 implementation
  ➢ Limited number of output display routes
    • A, X, Y, and flags
  ➢ SignalTap II Logic Analyzer
    • Real time.
    • Slower clock used for debug.
Debugging

• Common bugs
  ➢ SUMS, I_ADDC not turned off after next cycle
  ➢ PC incrementing at wrong cycle
  ➢ Resetting the initialization ROM

Surprisingly few errors regarding:
  ➢ Mask overlap
  ➢ Edge-triggered / level-sensitive misbehavior
  ➢ Mealy machine
=> Our ~500 timing diagrams did pay off!
Bouncing Ball

• In code: 8 statements

Define: X, Y, sizeX, sizeY, dirX, dirY

If X = sizeX
    dirX = 0;
end
If X = 0
    dirX = 1;
end
while clock and dirX=0
    X = X-1;
end
while clock and dirX=1
    X = X+1;
end

If Y = sizeY
    dirY = 0;
end
If Y = 0
    dirY = 1;
end
while clock and dirY=0
    Y = Y-1;
end
while clock and dirY=1
    Y = Y+1;
end
**Bouncing Ball**

- **In assembly language: 12 branches, 50 lines**

<table>
<thead>
<tr>
<th>Branch</th>
<th>Assembly Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>--INIT</strong></td>
<td>LDX #$00 A2 ( \text{INX} ) E8 LDA #$00 A9</td>
</tr>
<tr>
<td></td>
<td>LDY #$00 A0 CPX $0070 EC STA $0072 8D</td>
</tr>
<tr>
<td></td>
<td>LDA #$06 --sizeX A9 BEQ (BRC7) F0 JMP (BRC2) 4C</td>
</tr>
<tr>
<td></td>
<td>STA $0070 8D BNE (BRC2)-(BRC12) D0</td>
</tr>
<tr>
<td></td>
<td>LDA #$04 --sizeY A9</td>
</tr>
<tr>
<td></td>
<td>STA $0071 8D (--\text{BRC4})</td>
</tr>
<tr>
<td></td>
<td>LDA #$01 --dirX A9 DEX CA STA $0072 8D</td>
</tr>
<tr>
<td></td>
<td>STA $0072 8D CPX #$00 E0 JMP (BRC2) 4C</td>
</tr>
<tr>
<td></td>
<td>LDA #$01 --dirY A9 BEQ (BRC8) F0</td>
</tr>
<tr>
<td></td>
<td>STA $0073 8D BNE (BRC2)-(BRC12) D0 (--\text{BRC9})</td>
</tr>
<tr>
<td><strong>--BRC1</strong></td>
<td>LDA $0072 AD ( \text{INY} ) C8 JMP (BRC1) 4C</td>
</tr>
<tr>
<td></td>
<td>CMP #$01 C9 CPY $0071 CC</td>
</tr>
<tr>
<td></td>
<td>BEQ (BRC3) F0 BEQ (BRC9) F0 (--\text{BRC10})</td>
</tr>
<tr>
<td></td>
<td>BNE (BRC4) D0 BNE (BRC1)-(BRC11) D0 LDA #$00 A9</td>
</tr>
<tr>
<td><strong>--BRC2</strong></td>
<td>LDA $0073 AD ( \text{DEY} ) 88</td>
</tr>
<tr>
<td></td>
<td>CMP #$01 C9 CPY #$00 C0 JMP (BRC1) 4C</td>
</tr>
<tr>
<td></td>
<td>BEQ (BRC5) F0 BEQ (BRC10) F0 (--\text{BRC12})</td>
</tr>
<tr>
<td></td>
<td>BNE (BRC6) D0 BNE (BRC1)-(BRC11) D0</td>
</tr>
<tr>
<td><strong>--BRC3</strong></td>
<td>LDA #$00 A9</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC7})</td>
</tr>
<tr>
<td></td>
<td>LDA $0072 8D</td>
</tr>
<tr>
<td><strong>--BRC4</strong></td>
<td>LDA #$01 A9</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC8})</td>
</tr>
<tr>
<td><strong>--BRC5</strong></td>
<td>LDA $0073 8D</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC9})</td>
</tr>
<tr>
<td><strong>--BRC6</strong></td>
<td>LDA #$00 A9</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC10})</td>
</tr>
<tr>
<td><strong>--BRC7</strong></td>
<td>LDA $0072 8D</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC11})</td>
</tr>
<tr>
<td><strong>--BRC8</strong></td>
<td>LDA #$01 A9</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC12})</td>
</tr>
<tr>
<td><strong>--BRC9</strong></td>
<td>LDA $0073 8D</td>
</tr>
<tr>
<td><strong>--BRC10</strong></td>
<td>LDA #$01 A9</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC13})</td>
</tr>
<tr>
<td><strong>--BRC11</strong></td>
<td>LDA $0073 8D</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC14})</td>
</tr>
<tr>
<td><strong>--BRC12</strong></td>
<td>LDA #$00 A9</td>
</tr>
<tr>
<td></td>
<td>(--\text{BRC15})</td>
</tr>
</tbody>
</table>
Bouncing Ball

- In the ROM: 98 bytes of ROM