RUBIK'S CUBE SOLVER

Embedded Systems Design Project

Heather Fisher hlf2119
Yin-Chieh Wang yw2491
Zongheng Wang zw2223
Ifeoma Okereke iro2103

Project Overview

- Solve Rubik's Cubes with ease
- Displays each move
- Uses Thistlethwaite’s algorithm
- Speed up with FPGA
- How will software and hardware work to solve a Rubik's Cube?

Project Design

- VGA Display
  - Pseudo 3D Model
  - Easy to understand
- Keyboard
  - PS2 Keyboard
- Acceleration Block
  - Use hardware to speed up 3 most frequently called functions

Design for the VGA Display

- 1st Implementation
  - 6 Faces of Cube
- 2nd Implementation
  - 2 Pseudo 3D cubes
  - View from front and back
- 3rd Implementation
  - Single 3D cube
  - Uses extending lines
  - Projects surface

Keyboard

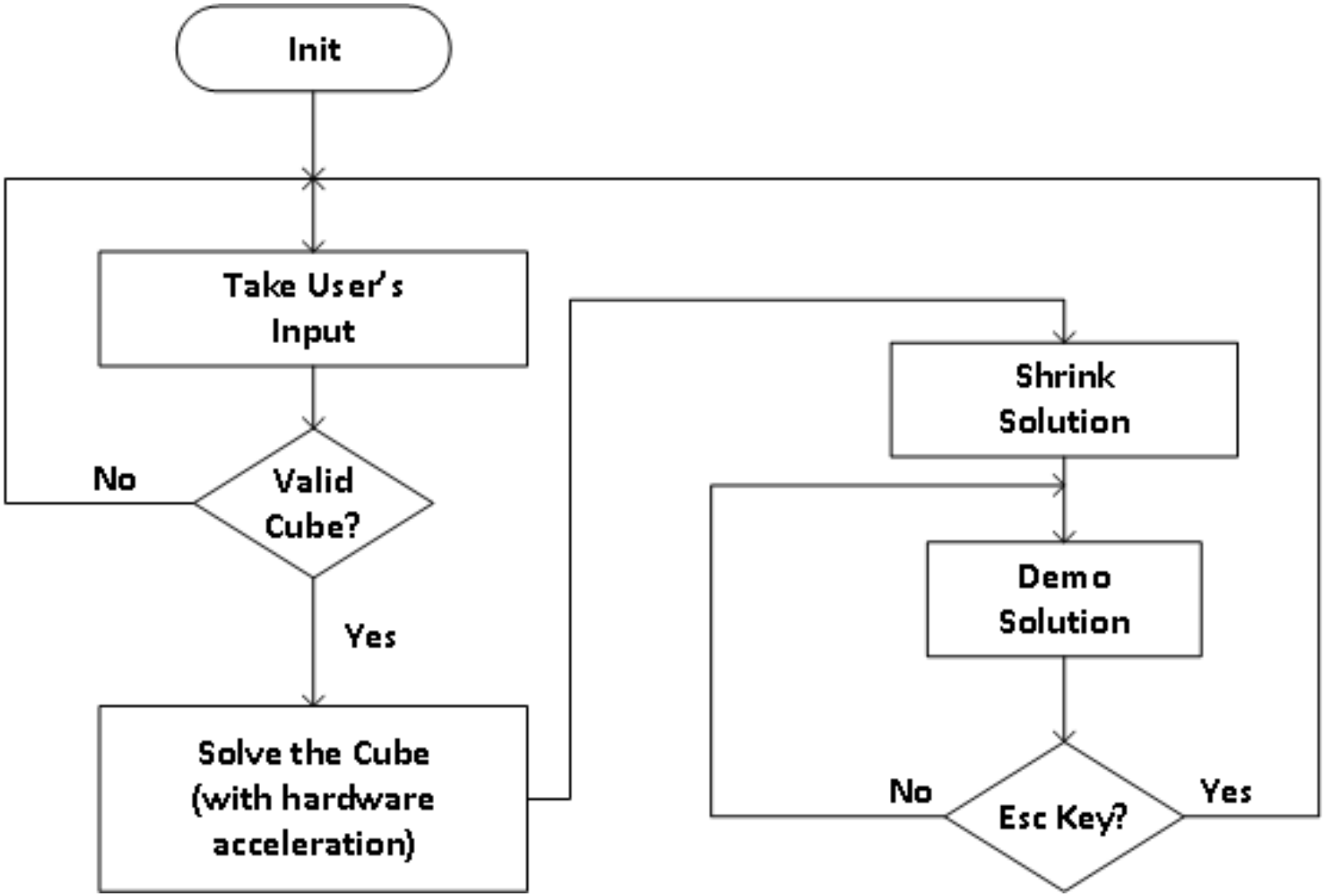
- Uses Numpad
  - Each number represents a cubie
  - Press key to "scroll" through colors
- Tab Key
  - Change selected side of cube
- Enter Key
  - Solve the cube
- Arrow Keys
  - Left/Right get instructions for solving
- Escape
  - Restart

Software - gprof result of the Thistlethwaite's Algorithm

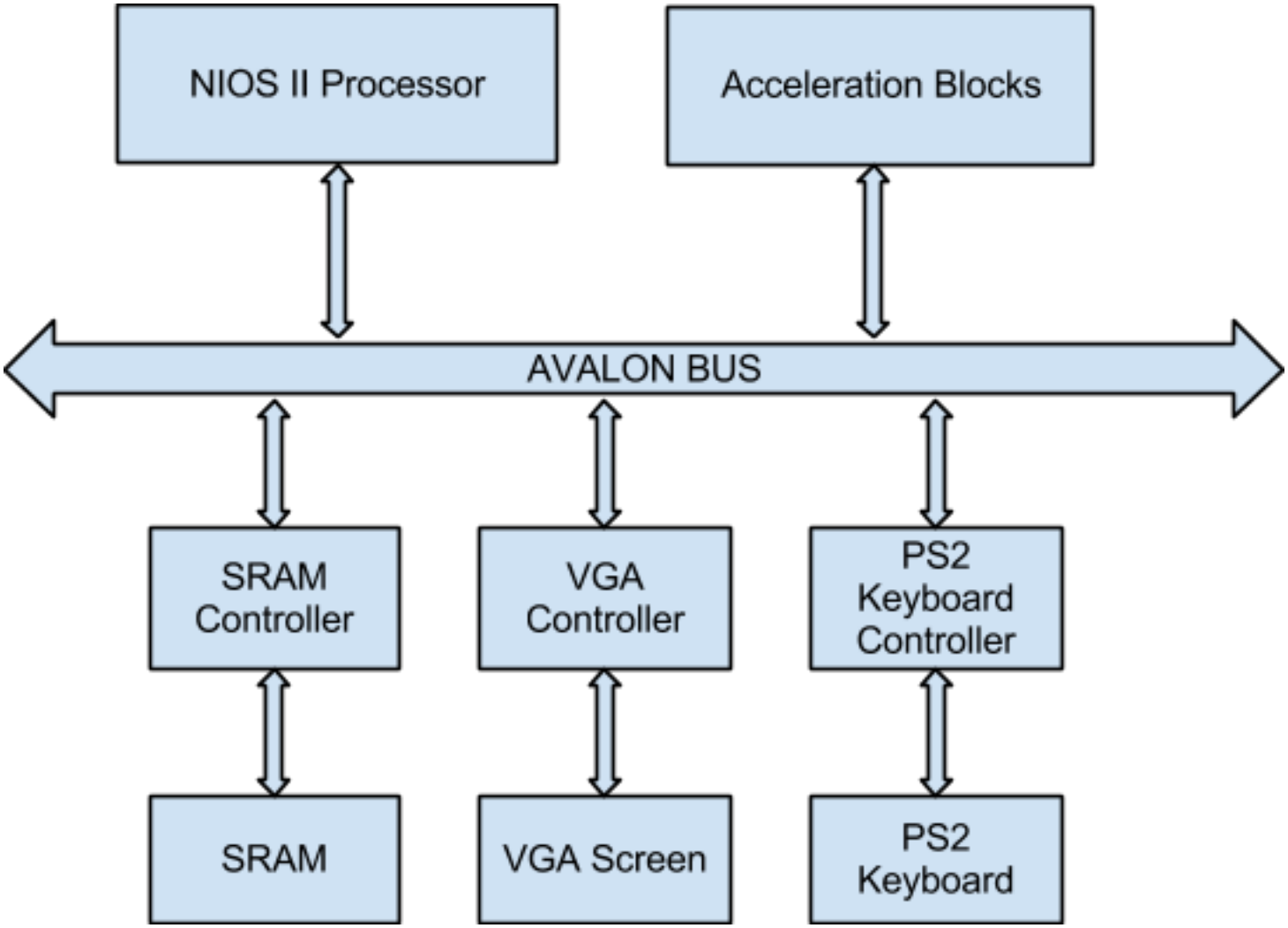
% Time	Cumulative Seconds	Self Seconds	Calls	Self ms/Call	Total ms/Call	name
64.41	0.18	0.18	2646168	0.00	0.00	cycle
14.31	0.22	0.04	794597	0.00	0.00	permtonum
10.73	0.25	0.03	2675292	0.00	0.00	twist
7.16	0.27	0.02	477227	0.00	0.00	getposition
3.58	0.28	0.01	42624	0.00	0.00	numtoperm
0.00	0.28	0.00	661542	0.00	0.00	domove
0.00	0.28	0.00	19629	0.00	0.00	reset
0.00	0.28	0.00	19621	0.00	0.00	setposition
0.00	0.28	0.00	33	0.00	2.31	searchphase
0.00	0.28	0.00	8	0.00	25.53	filltable

*Each sample counts as 0.01

Software - RCS Flow Chart



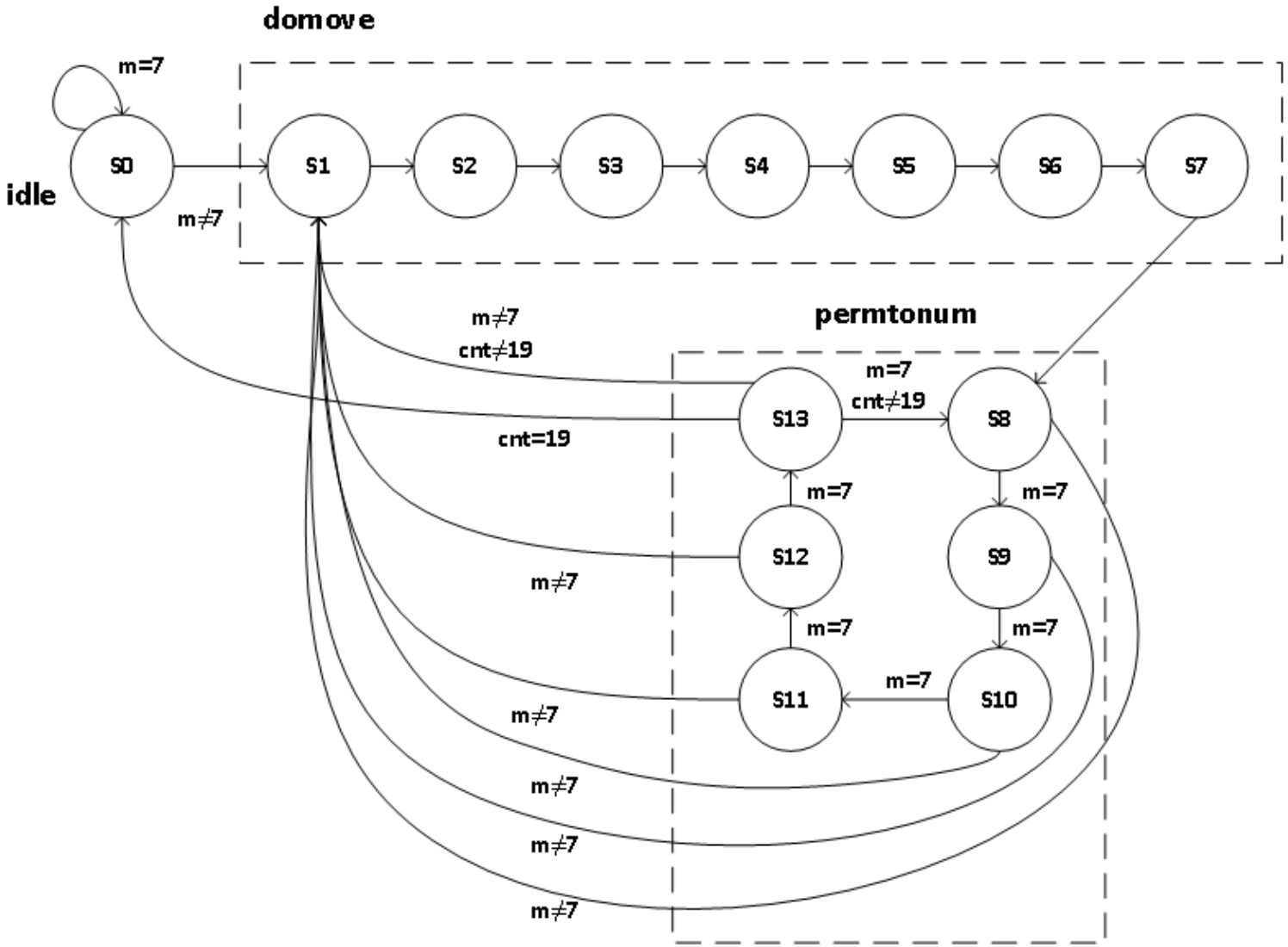
Hardware - Overview



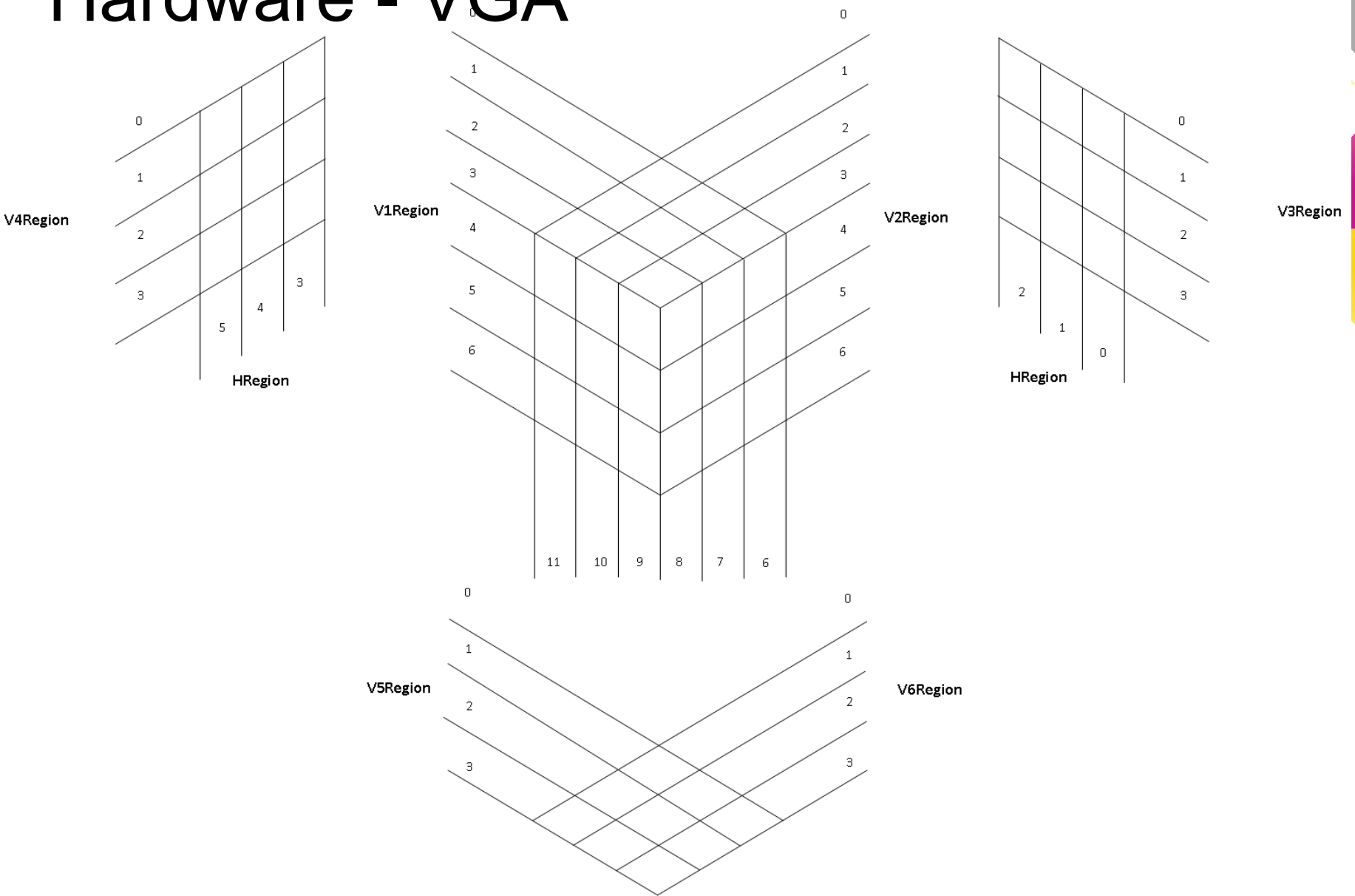
Hardware - Acceleration Blocks

- Objective: Modifying data in a true_dual_port_ram_single_clock
- Port A: domove(cycle and twist), permtonum
- Port B: domove(cycle and twist), avalon bus interface

Hardware - Moore State Diagram



Hardware - VGA



Hardware - Text Display

- Define text display regions - rectangles
  - read the bitmap and display when falling in the regions

- Character bitmap
  - store the address of the top six bits in RAM
  - for the last three bits, use the last three bits of Hcount

```
000   : 00011000 ; %     **       %
001   : 00111100 ; %    ****      %
002   : 01100110 ; %   **  **     %
003   : 01111110 ; %   ******     %
004   : 01100110 ; %   **  **     %
005   : 01100110 ; %   **  **     %
006   : 01100110 ; %   **  **     %
007   : 00000000 ; %              %
```

Experiences and Challenges

- Re-designed most of hardwares multiple times
  - painful but worthwhile
  - acceleration blocks - 3 times; VGA controller - 4 times; software - 2 times;
  - simple is good - efficient and flexible

- Sometimes printf doesn't work and downloading ELF file gives an error
  - usually means bad hardware design

Lessons Learned

- VHDL is not like C and Java
- Design before you code
  - Block Diagram
  - Timing Diagram
- If you have 1000s of line of VHDL code in one file...Something is wrong

Summary

- It solves the cube!
- User friendly
- More efficient

On FPGA Without Acceleration	80-100 sec*
On FPGA with Acceleration	5-15 sec*
On computer	10-20 msec*

*time depends on difficulty of the cube