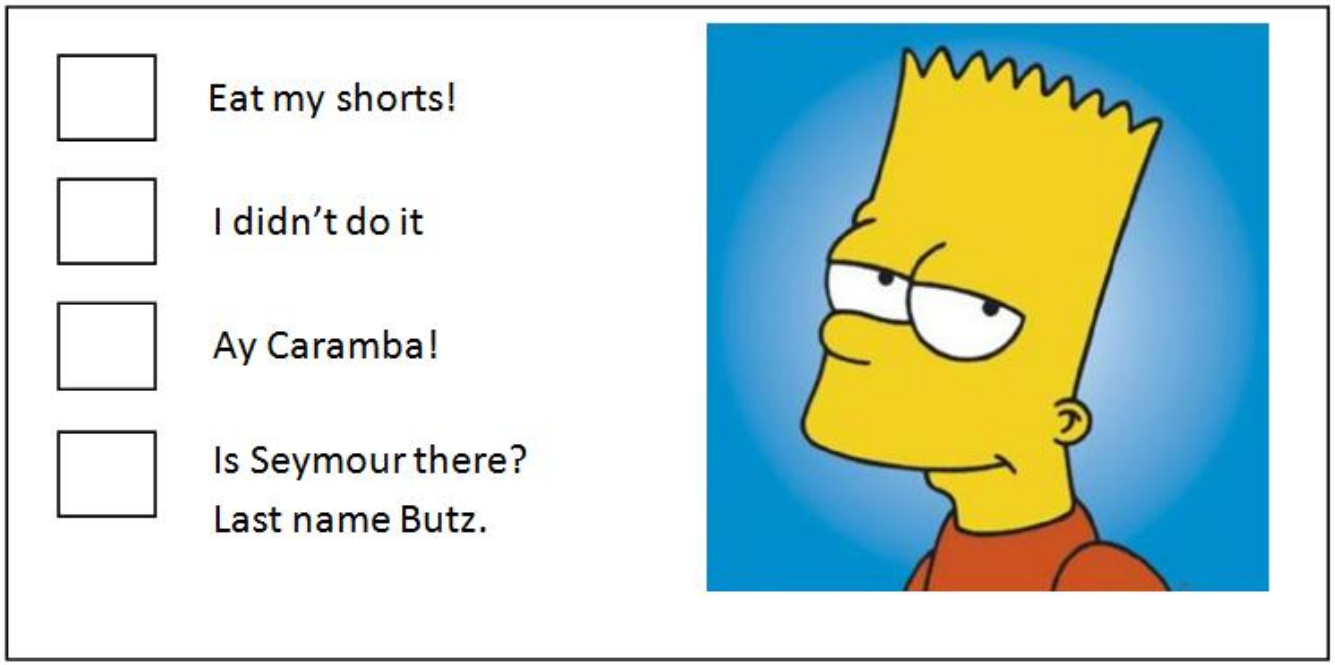
Funny Sound Board

E4840 Spring 2011 Final Project

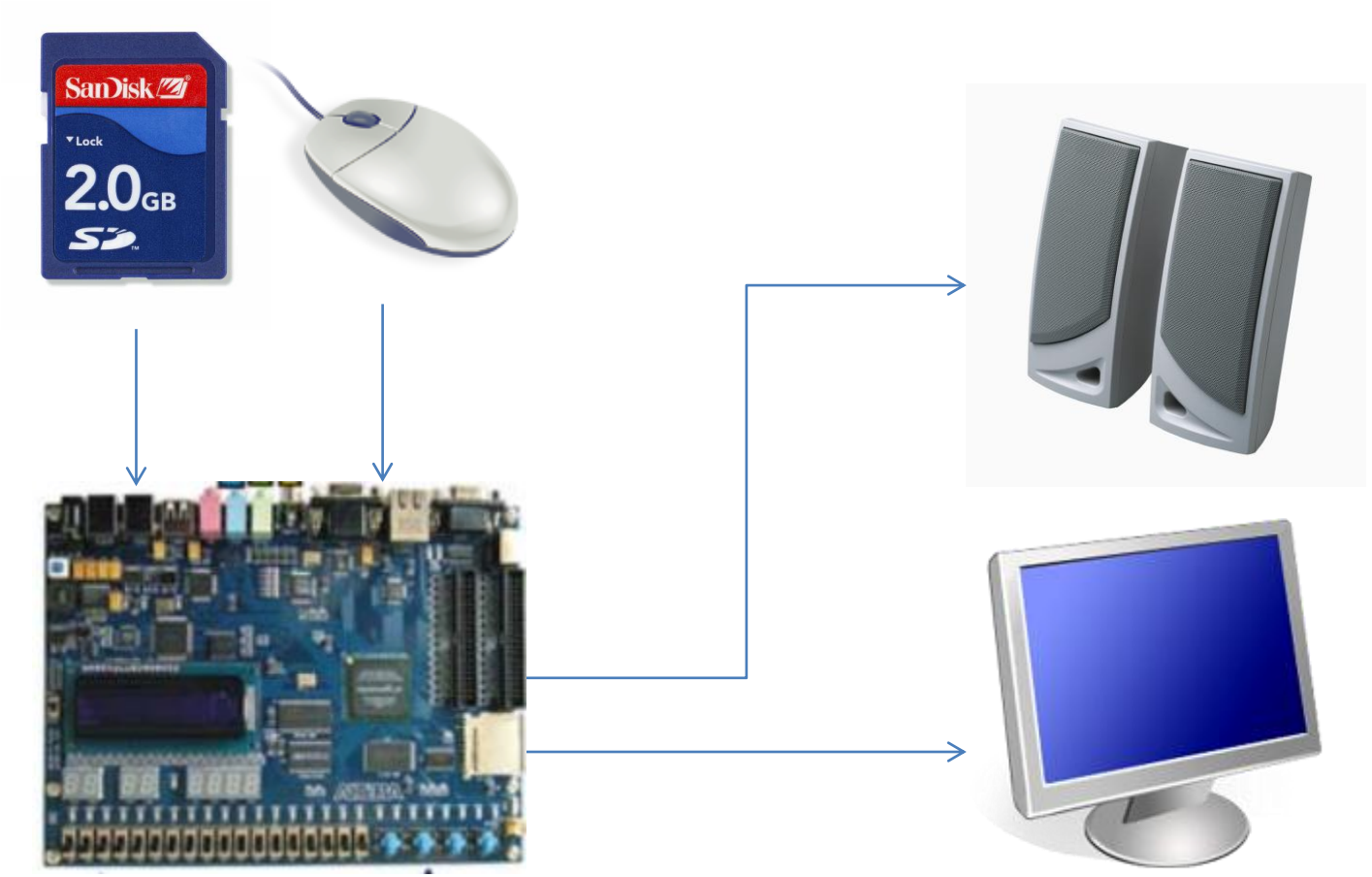
Dawie Liu & Lee Zhu

Goal was to make a project that generate laughs

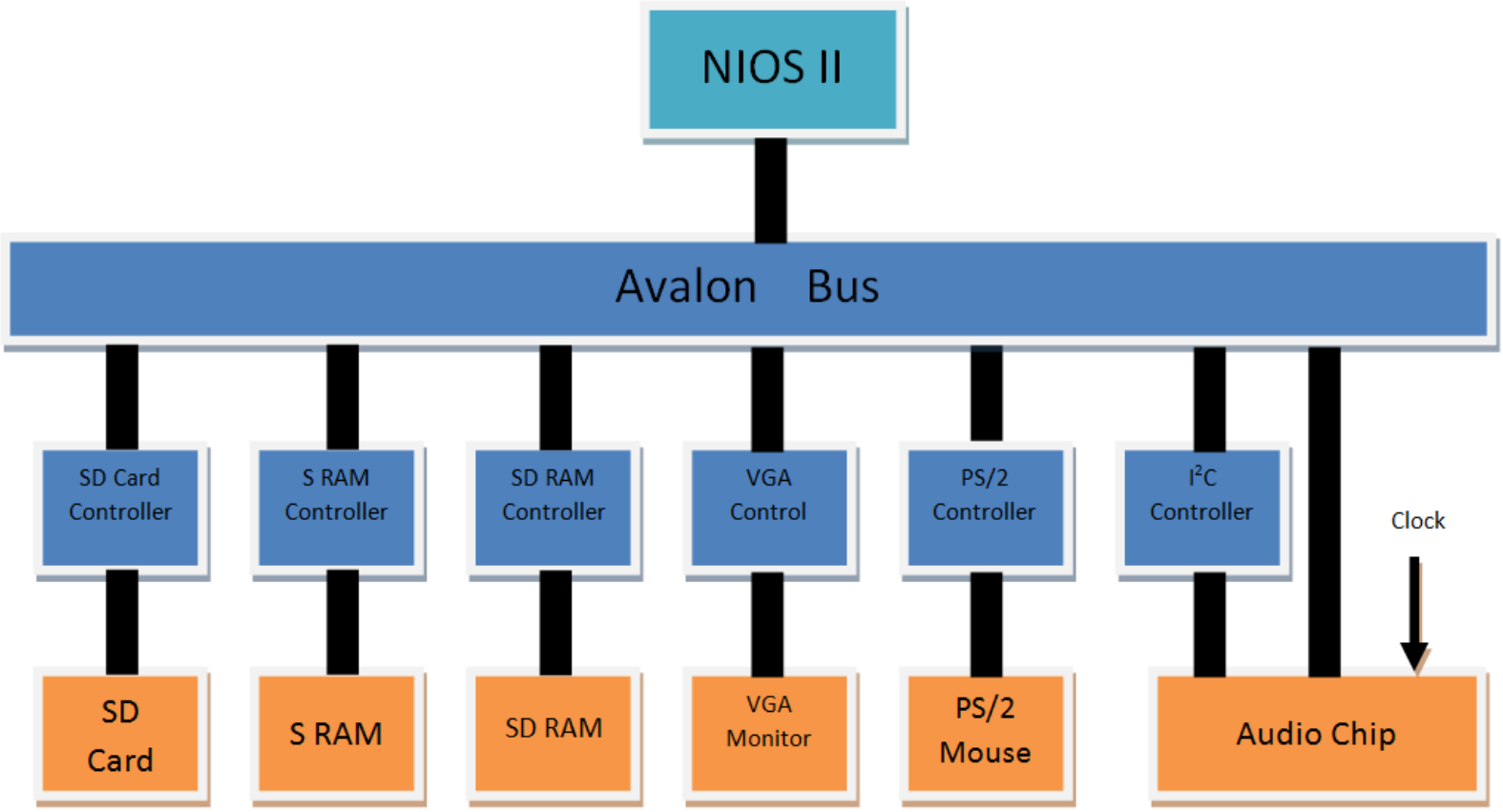


- Click a button to play sound
- Put it at place where people wait

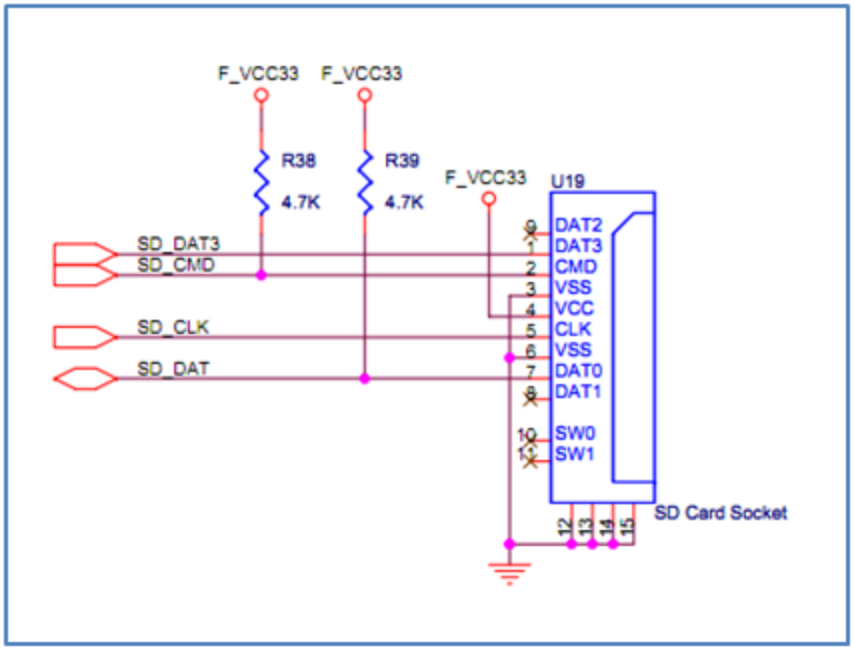
Use DE2 board to make this possible



Block diagram



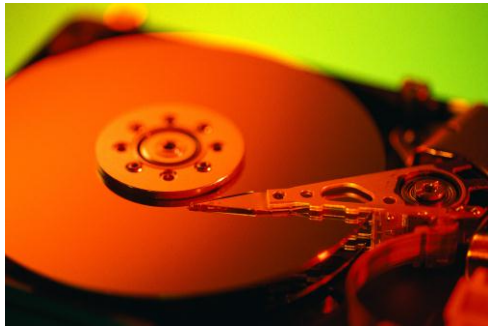
SD card interface is through software



- Altera supplies example driver
- We were able to optimize it a bit
- Hardware implementation would be much faster

Files are stored on SD card just like on a hard drive

- FAT16 software is ported from external source[1]
- We used a very simple file system
- Assume no fragmentation and read only



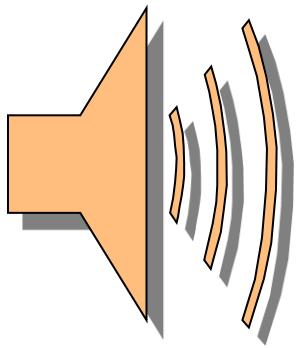
[1] copyright © 2000 - 2011, Glen George.

Mouse is also software driven

- Similar to keyboard from Lecture 3
- Need to translate mouse data points into smooth motion
- We used the bouncing ball method from lab3 to interface the mouse with the VGA controller

Audio is complicated on the DE2

- Make sure I2C initialization is correct
- Set the correct clock frequency and modes.
- Complicated issue! Whole masters thesis have been spent on just this interface alone!

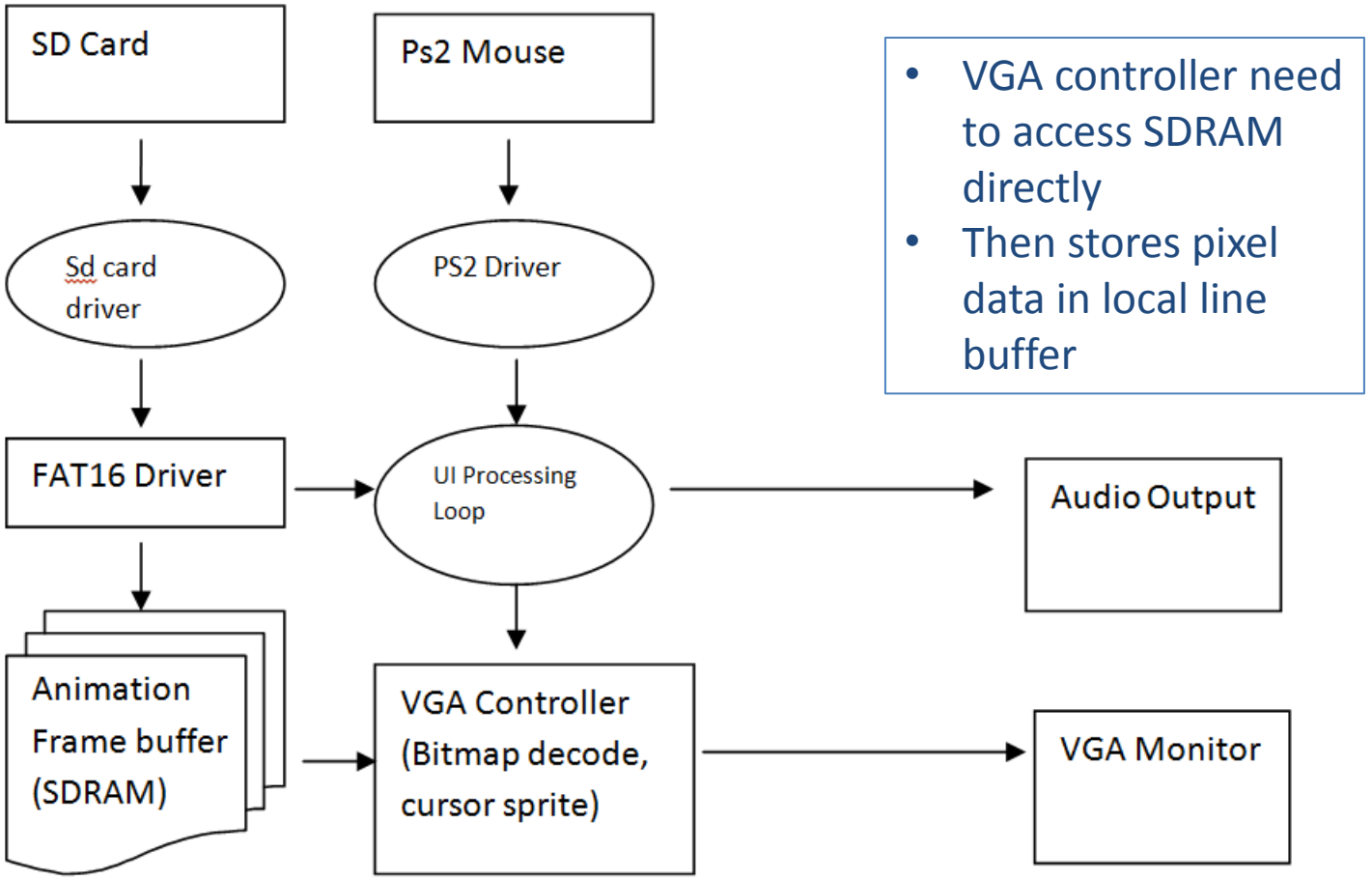


We wanted large pictures and video-like animation

- Need to display multiple high-resolution color frames
- 640 x 480 x 16 bits/pixel => 600 KB/Frame

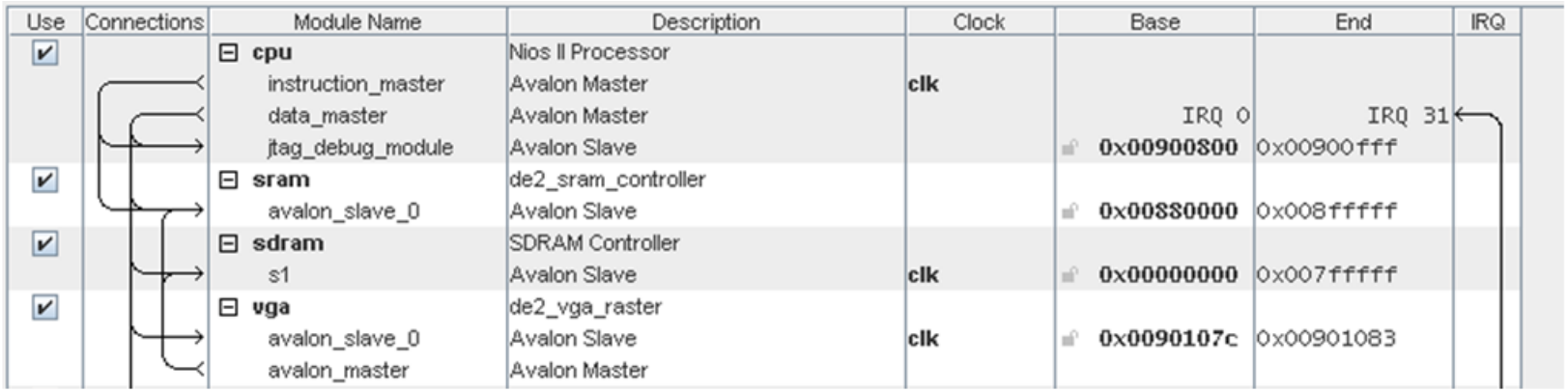
- Do not have enough memory on board
- Did not have resource to implement decompression (JPEG, MPEG, etc)

Solution is to use SDRAM as frame buffer



- VGA controller need to access SDRAM directly
- Then stores pixel data in local line buffer

Add VGA component as Avalon Master



Use	Connections	Module Name	Description	Clock	Base	End	IRQ
✔		cpu	Nios II Processor				
		instruction_master	Avalon Master	clk			
		data_master	Avalon Master			IRQ 0	IRQ 31
		jtag_debug_module	Avalon Slave		0x00900800	0x00900fff	
✔		sram	de2_sram_controller				
		avalon_slave_0	Avalon Slave		0x00800000	0x008fffff	
✔		sdram	SDRAM Controller				
		s1	Avalon Slave	clk	0x00000000	0x007fffff	
✔		vga	de2_vga_raster				
		avalon_slave_0	Avalon Slave	clk	0x0090107c	0x00901083	
		avalon_master	Avalon Master				

chipselect	avalon_slave_0
write	avalon_slave_0
address	avalon_slave_0
writedata	avalon_slave_0
avm_read_master_read	avalon_master
avm_read_master_address	avalon_master
avm_read_master_readdata	avalon_master
avm_read_master_waitrequest	avalon_master

- Use SOPC
- Implement state machine to read memory

Pitfalls to avoid – bus width issues

Dynamic Bus Sizing Master-to-Slave Address Mapping

Master Byte Address *(1)*	32-Bit Master Data		
	When Accessing an 8-Bit Slave Port	When Accessing a 16-Bit Slave Port	When Accessing a 64-Bit Slave Port
0x00	OFFSET[3]$_{7..0}$:OFFSET[2]$_{7..0}$: OFFSET[1]$_{7..0}$:OFFSET[0]$_{7..0}$	OFFSET[1]$_{15..0}$:OFFSET[0]$_{15..0}$ *(2)*	OFFSET[0]$_{31..0}$
0x04	OFFSET[7]$_{7..0}$:OFFSET[6]$_{7..0}$: OFFSET[5]$_{7..0}$:OFFSET[4]$_{7..0}$	OFFSET[3]$_{15..0}$:OFFSET[2]$_{15..0}$	OFFSET[0]$_{63..32}$
0x08	OFFSET[11]$_{7..0}$:OFFSET[10]$_{7..0}$: OFFSET[9]$_{7..0}$:OFFSET[8]$_{7..0}$	OFFSET[5]$_{15..0}$:OFFSET[4]$_{15..0}$	OFFSET[1]$_{31..0}$
0x0C	OFFSET[15]$_{7..0}$:OFFSET[14]$_{7..0}$: OFFSET[13]$_{7..0}$:OFFSET[12]$_{7..0}$	OFFSET[7]$_{15..0}$:OFFSET[6]$_{15..0}$	OFFSET[1]$_{63..32}$
. . .		. . .	. . .

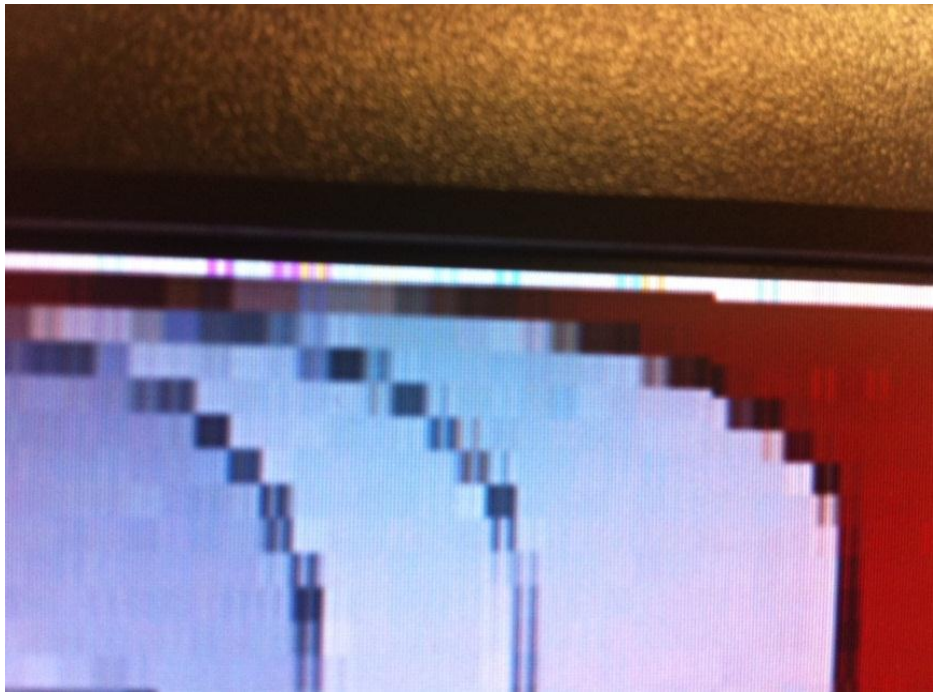
Notes to Table 3–3:

(1) Although the master is issuing byte addresses, it is accessing full 32-bit words.

(2) For all slave entries, [<*n*>] is the word offset and the subscript values are the bits in the word.

- Make sure words are aligned to ensure bug-free
- Use DWORD read/write to avoid waste

Performance of VGA is bad due to SDRAM latency



- SDRAM in random access mode can only supply ~100 pixels per line in real time
- Pixilation and raster line skipping as a result
- Effectively operating at reduced resolution

Need to operate in pipeline or burst mode! => More complicated state machine

Audio playback was noisy and slightly slow

- Maybe due to setup of codec chip and audio core
- Could be from our source files which were up-sampled to 48KHz
- We sent data to codec directly from SD card, perhaps buffering could improve performance

Thanks to Dr. Edwards for an enlightening semester of class!