



# Lab 8: SD Card Reader Circuit

Chun-Jen Tsai and Lan-Da Van  
Department of Computer Science  
National Yang Ming Chiao Tung University  
Taiwan, R.O.C.  
*Fall, 2022*



# Lab 8: SD Card Reader Circuit

Lab 8

- ◆ In this lab, you will design a circuit to read a text file from an SD card, and count the number of three-letter words in the file.
  - Example of three-letter words: “the”, “and”, “all”, ...
  - The number of the three-letter words will be displayed on the 1602 LCD screen.
  
- ◆ The lab file submission deadline is on 11/28 by 6:00pm.





# SD Card Specification (1/4)

Lab 8



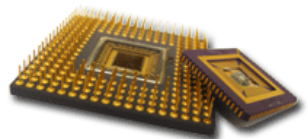


# SD Card Specification (2/4)

Lab 8

	圖示	檔案系統	容量
SD		FAT12, FAT16	上限 2 GB
SDHC		FAT32	4GB ~ 32GB
SDXC		exFAT	32GB ~ 2TB
SDUC		exFAT	2TB ~ 128TB


SD = **S**ecure **D**igital  
 SDHC = **S**ecure **D**igital **H**igh **C**apacity  
 SDXC = **S**ecure **D**igital **e**Xtended **C**apacity  
 SDUC = **S**ecure **D**igital **U**ltra **C**apacity

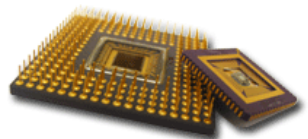




# SD Card Specification (3/4)

Lab 8

最低寫入速度	速度等級 (Speed Class)	UHS 速度等級 (UHS Speed Class)	影片速度等級 (Video Speed Class)	適用拍攝影片
2 MB/s	 Class 2 (C2)	-	-	720p 影片
4 MB/s	 Class 4 (C4)	-	-	720p 影片
6 MB/s	 Class 6 (C6)	-	<b>V6</b> Class 6 (V6)	720p 影片
10 MB/s	 Class 10 (C10)	 Class 1 (U1)	<b>V10</b> Class 10 (V10)	1080p 影片
30 MB/s	-	 Class 3 (U3)	<b>V30</b> Class 30 (V30)	1080p 影片 60/120 fps
60 MB/s	-	-	<b>V60</b> Class 60 (V60)	4K 影片 60/120 fps
90 MB/s	-	-	<b>V90</b> Class 90 (V90)	8K 影片 60/120 fps





# SD Card Specification (4/4)

Lab 8

- ◆ The SD card that we use follows the secure digital high capacity (SDHC) standard, formatted with the FAT32 file system.
- ◆ The logical structure is composed of 512-byte blocks, starting at block number 0.
  - An 8GB SD card will be used in the lab.
- ◆ SD cards support at least two different I/O interfaces. In this lab, we use the serial Serial Peripheral Interconnect (SPI) interface to read data.





# SD Card I/O Interface (1/2)

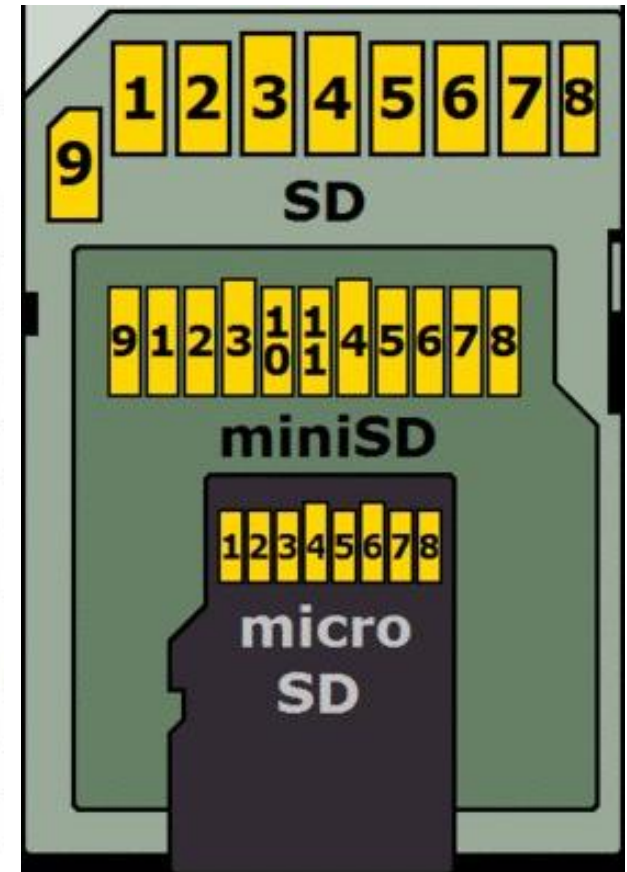
Lab 8

◆ An SDHC card has three different operation modes:

- SPI mode
- One-bit SD bus mode
- Four-bit SD bus mode

SPI Bus Mode

MMC Pin	SD Pin	miniSD Pin	microSD Pin	Name	I/O	Logic	Description
1	1	1	2	nCS	I	PP	SPI Card Select [CS] (Negative logic)
2	2	2	3	DI	I	PP	SPI Serial Data In [MOSI]
3	3	3		VSS	S	S	Ground
4	4	4	4	VDD	S	S	Power
5	5	5	5	CLK	I	PP	SPI Serial Clock [SCLK]
6	6	6	6	VSS	S	S	Ground
7	7	7	7	DO	O	PP	SPI Serial Data Out [MISO]
	8	8	8	NC nIRQ	. O	. OD	Unused (memory cards) Interrupt (SDIO cards) (Negative logic)
	9	9	1	NC	.	.	Unused
		10		NC	.	.	Reserved
		11		NC	.	.	Reserved







# SD Card I/O Interface (2/2)

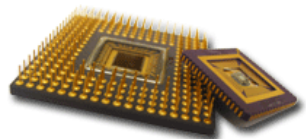
Lab 8

One-Bit SD Bus Mode

MMC Pin	SD Pin	miniSD Pin	microSD Pin	Name	I/O	Logic	Description
1	1	1	2	CD	I/O	.	Card Detection (by host), and Non-SPI Mode Detection (by card)
2	2	2	3	CMD	I/O	PP, OD	Command, Response
3	3	3		VSS	S	S	Ground
4	4	4	4	VDD	S	S	Power
5	5	5	5	CLK	I	PP	Serial clock
6	6	6	6	VSS	S	S	Ground
7	7	7	7	DAT0	I/O	PP	SD Serial Data 0
	8	8	8	NC nIRQ	.	OD	Unused (memory cards) Interrupt (SDIO cards) (Negative Logic)
	9	9	1	NC	.	.	Unused
		10		NC	.	.	Reserved
		11		NC	.	.	Reserved

Four-Bit SD Bus Mode

MMC Pin	SD Pin	miniSD Pin	microSD Pin	Name	I/O	Logic	Description
.	1	1	2	DAT3	I/O	PP	SD Serial Data 3
.	2	2	3	CMD	I/O	PP, OD	Command, Response
.	3	3		VSS	S	S	Ground
.	4	4	4	VDD	S	S	Power
.	5	5	5	CLK	I	PP	Serial clock
.	6	6	6	VSS	S	S	Ground
.	7	7	7	DAT0	I/O	PP	SD Serial Data 0
	8	8	8	DAT1 nIRQ	I/O	PP	SD Serial Data 1 (memory cards) Interrupt Period (SDIO cards share pin via protocol)
	9	9	1	DAT2	I/O	PP	SD Serial Data 2
		10		NC	.	.	Reserved
		11		NC	.	.	Reserved



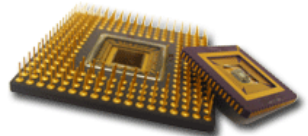




# SD Card Initialization

Lab 8

- ◆ During the initialization phase, the SD card controller negotiates with the card to determine which type of card is used: SD, SDHC, SDXC, and SDUC.
  - The controller uses a slower clock (500kHz) to talk to the SD card during the negotiation phase.
- ◆ Once the card is initialized, the SD card controller can use a faster clock (e.g., the system clock) for read/write operations, as long as the card can handle the speed.

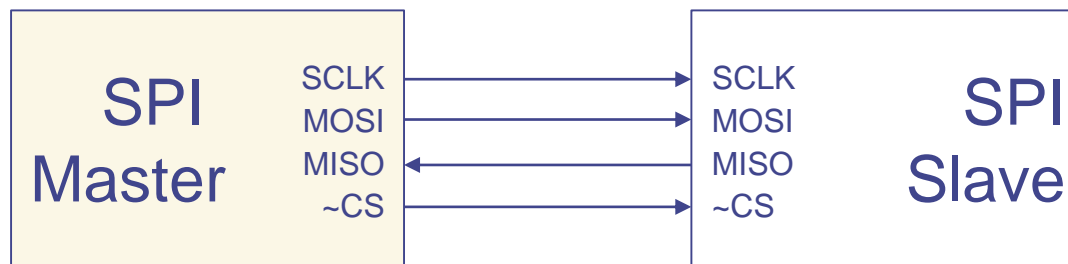




# Serial Peripheral Interconnect (SPI)

Lab 8

- ◆ SPI is introduced by Motorola in 1980's for their MCU.
  - Short-distance synchronous serial communications for SD cards, LCDs screens, audio codecs, boot flash, etc.
  - A four-wire, full-duplex, master-slave serial bus
  - One master, multiple slaves
  - Open-loop transmission, no slave acknowledgement protocol

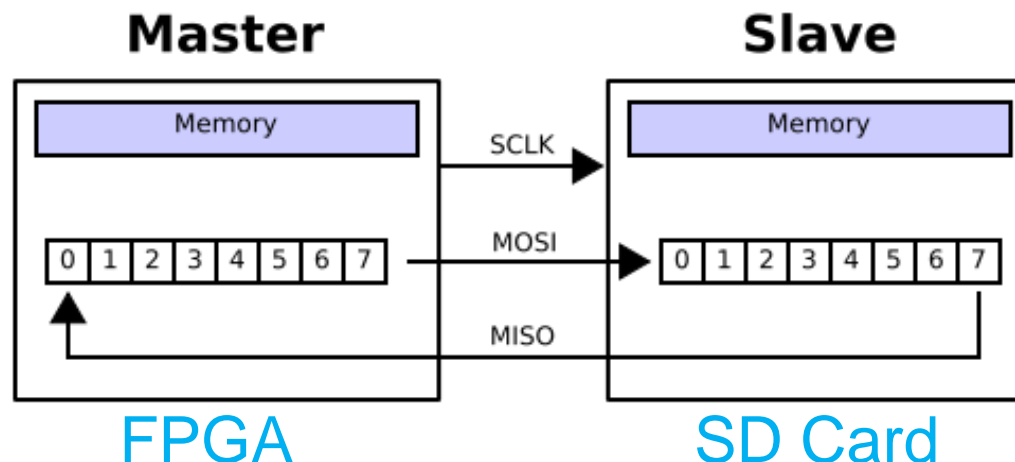




# SPI Data Communication

Lab 8

- ◆ The master selects the target slave via the CS pin (microSD pin 2) first, then sends the clock signal to the slave.
- ◆ The master and slave exchange data one bit per clock cycle using shift registers.
  - Data sizes can be of 8-, 12-, or 16-bit, depending on the device.
  - The data sampling clock edge (rising or falling) also depends on the device → read data sheet of the device!

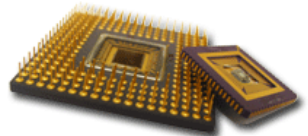




# Physical Structure and File Systems

Lab 8

- ◆ The logical structure of an SD card is simply composed of a sequence of 512-byte blocks.
  - Physically, SD cards are divided into 4KB ~ 32KB sectors.
  - 1 block = 512 Bytes
  - 1 sector = one or multiple blocks
- ◆ To create directories for file storage on the card, we must first partition the SD card and then format a logical file system on that partition.
- ◆ An SD card usually has one partition. However, it is possible to store multiple partitions and multiple file systems on a single SD card.

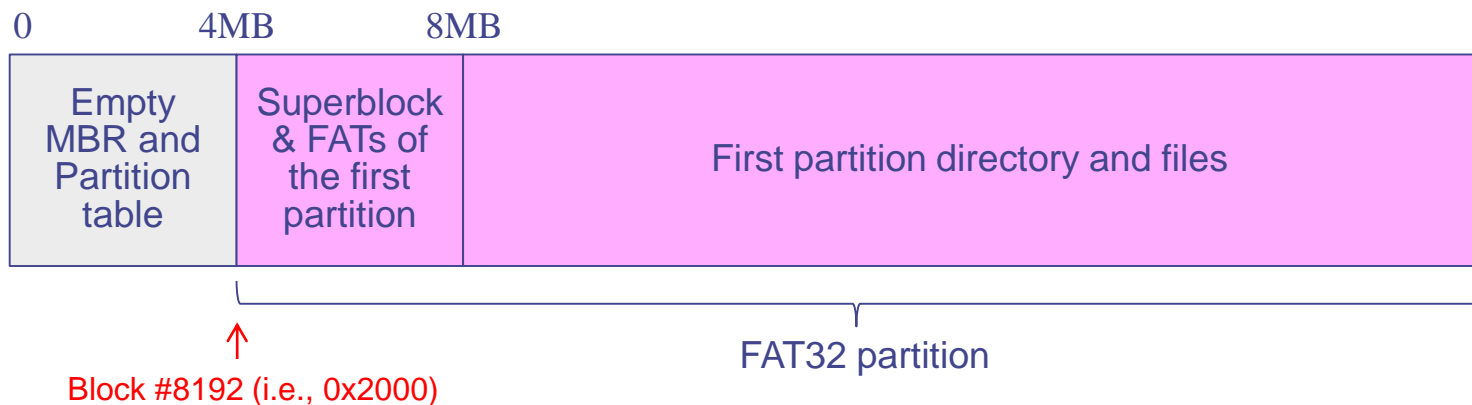




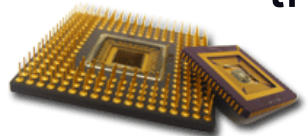
# Disk Partitions

Lab 8

- ◆ A physical disk can have several disk partitions, and each partition can be formatted to a file system.
- ◆ Typical partition structure of an SDHC card:



- ◆ If the card is bootable or has more than one partitions, then the Master Boot Record (MBR) and the partition table will not be empty.





# FAT32 Structure

Lab 8

- ◆ The FAT32 file system is a standard by Microsoft, and popularly used for mass storage devices.
- ◆ An FAT32 file system has the following components:
  - Boot sector: 512 bytes, possibly block#0, a.k.a. the super block
    - ◆ The boot sector contains information such as the size of the FAT, root directories, boot code, etc.
  - File allocation table (FAT): a table that shows which file is stored in which allocation units (each allocation unit is composed of several consecutive blocks); FAT basically contains many link lists of block numbers (one list per file).
  - Root directory: contains file names, file attributes, and the first allocation unit of all files in the root directory
  - Data area: the data blocks that actually store files



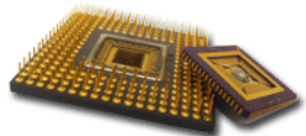
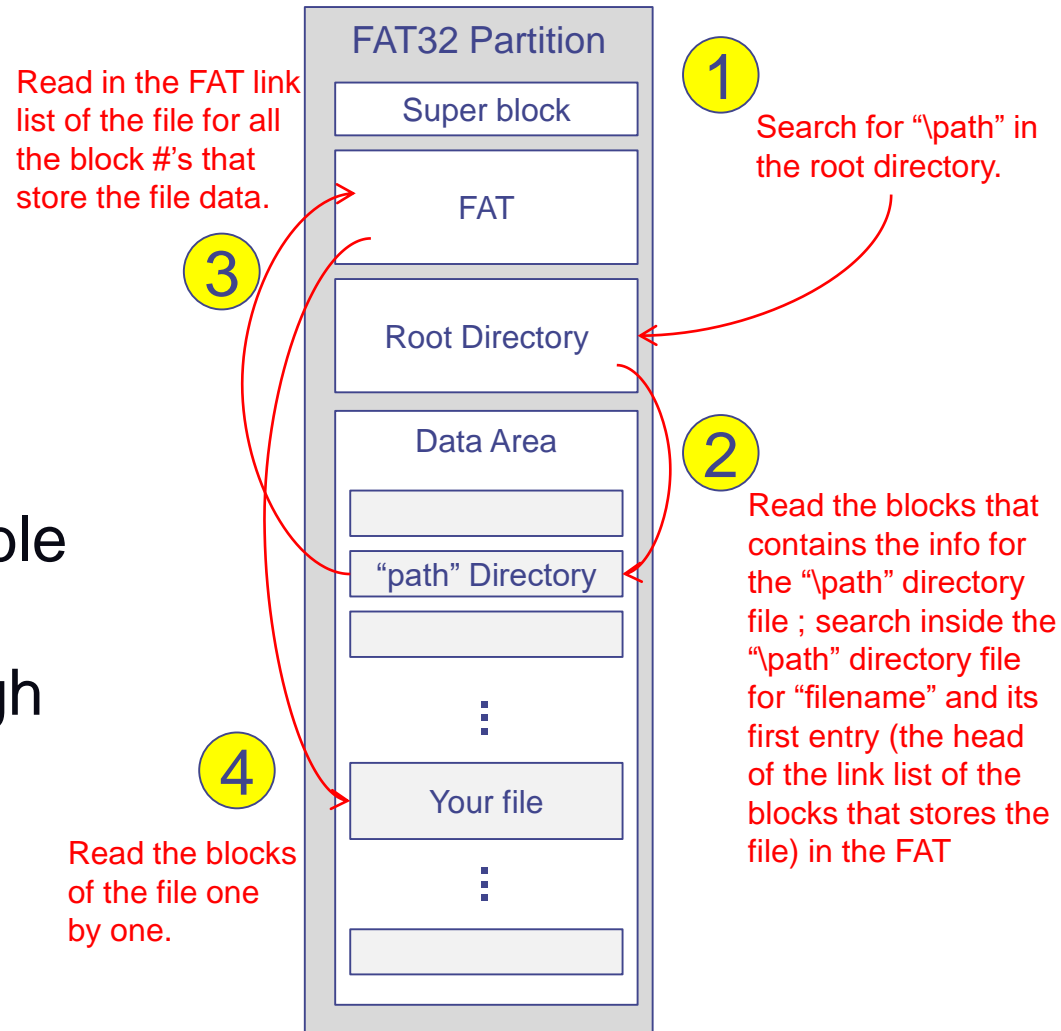




# File Structure of an FAT32 Partition

Lab 8

- ◆ To read a file of “\path\name” in an FAT32 file system, one shall follow the four steps shown in the figure.
- ◆ However, it is possible to read a **small** file without going through these steps!





# Sample Project Files of Lab 8

Lab 8

- ◆ The sample project files of Lab 8 has a top-level circuit, `lab8.v`, an SD card controller, `sd_card.v`, and other supporting files such as: `debounce.v`, `LCD_module.v`, `clk_divider.v`, and `lab8.xdc`.
- ◆ The circuit performs the following actions:
  - When the board is powered up, the SD card controller will initialize itself and enter a ready state.
  - When the user presses BTN2, the circuit reads a block of data from the SD card, and then print the first byte in the block on the LCD module.
  - Every time BTN2 is pressed, the next byte will be displayed.

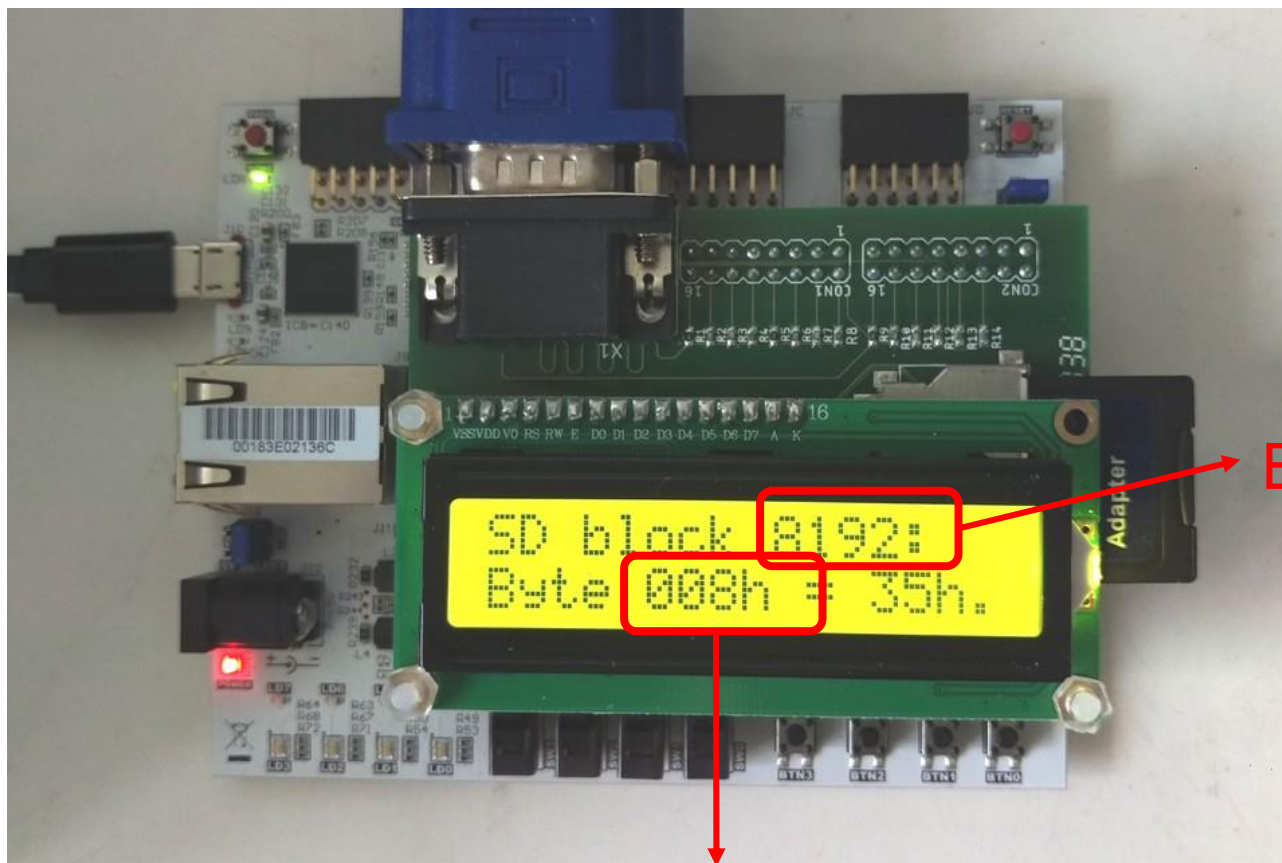




# Output of Lab 8 Sample Circuit

Lab 8

- ◆ For example, the following message is shown after 9 button hits (different card may show different result):



Address

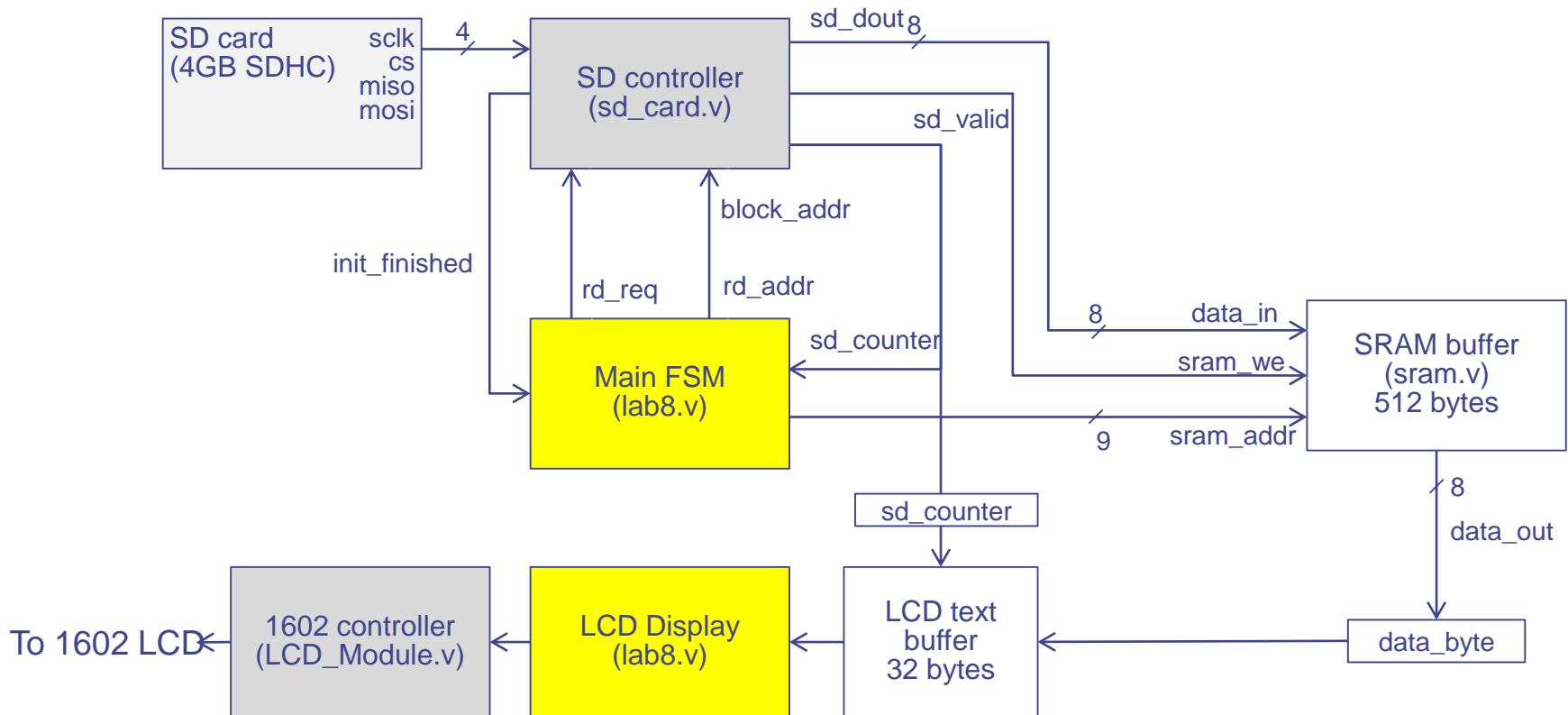
Block index



# System Diagram of the Sample Code

Lab 8

◆ The block diagram of the sample code of Lab 8:





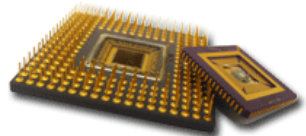
# SD Controller Signals

Lab 8

## ◆ The key signals that control SD card operations:

- `rd_req`: Triggers the reading of a block
- `block_addr[31:0]`: The block # of the SD card to read
- `init_finished`: SD card initialization is finished?
- `dout[7:0]`: Output one byte of data in the block per clock cycle.
- `sd_valid`: The output byte in `dout[7:0]` is ready

- ◆ If you set `rd_req` to 1 for one clock cycle, the SD card controller will read the data of the target block one byte at a time. Each time a data byte (of the 512 bytes) is ready, the flag `sd_valid` raises to 1 for one clock cycle.





# What You Need to Do for Lab 8

Lab 8

- ◆ You must design a circuit that:
- You can download “test.txt” from E3 and save it on the SD cards in the lab if it’s not there already.
  - The file begins with the word “DLAB\_TAG” and ends with the word “DLAB\_END”; each word is separated by white-spaces, punctuation marks or line feeds from the next word.
  - When BTN2 is pressed, the circuit reads the SD card and searches for an ASCII file, “test.txt”.
  - The circuit reads through the text file and counts the number of three-letter words in the text file, and prints the number to the 1602 LCD module using 4-digit hexadecimal format as follows:

**Found 007D words  
in the text file**







# Format of the Input Text File

Lab 8

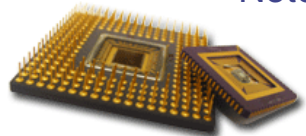
◆ The sample input text file, test.txt, is as follows.

```
DLAB_TAG↵
The Hitch Hiker's Guide to the Galaxy↵
↵
Far out in the uncharted backwaters of the unfashionable end of
the western spiral arm of the Galaxy lies a small unregarded
yellow sun.↵

. . . . .

But the story of this terrible, stupid Thursday, the story of its
extraordinary consequences, and the story of how these
consequences are inextricably intertwined with this remarkable
book begins very simply.↵
↵
It begins with a house.↵
DLAB_END↵
```

Note: ↵ stands for 0x0A.





# Layout of the File on the SD card

Lab 8

- ◆ A newly formatted SD card will have its initial files stored in consecutive 512-byte blocks.
- ◆ After some files are deleted, new files added may be stored in non-contiguous blocks.
- ◆ You can assume that test.txt is stored in consecutive blocks.

