

# COMPUTER FOUNDATIONS

COMP 597 – Computer Forensics



# Data Representation

# Data Representation

---

- Number System:

- Decimal

- Binary

- Hexadecimal

# Data Representation



01100010

# Data Representation



How do we represent the following decimal number in memory 555?

# Data Representation



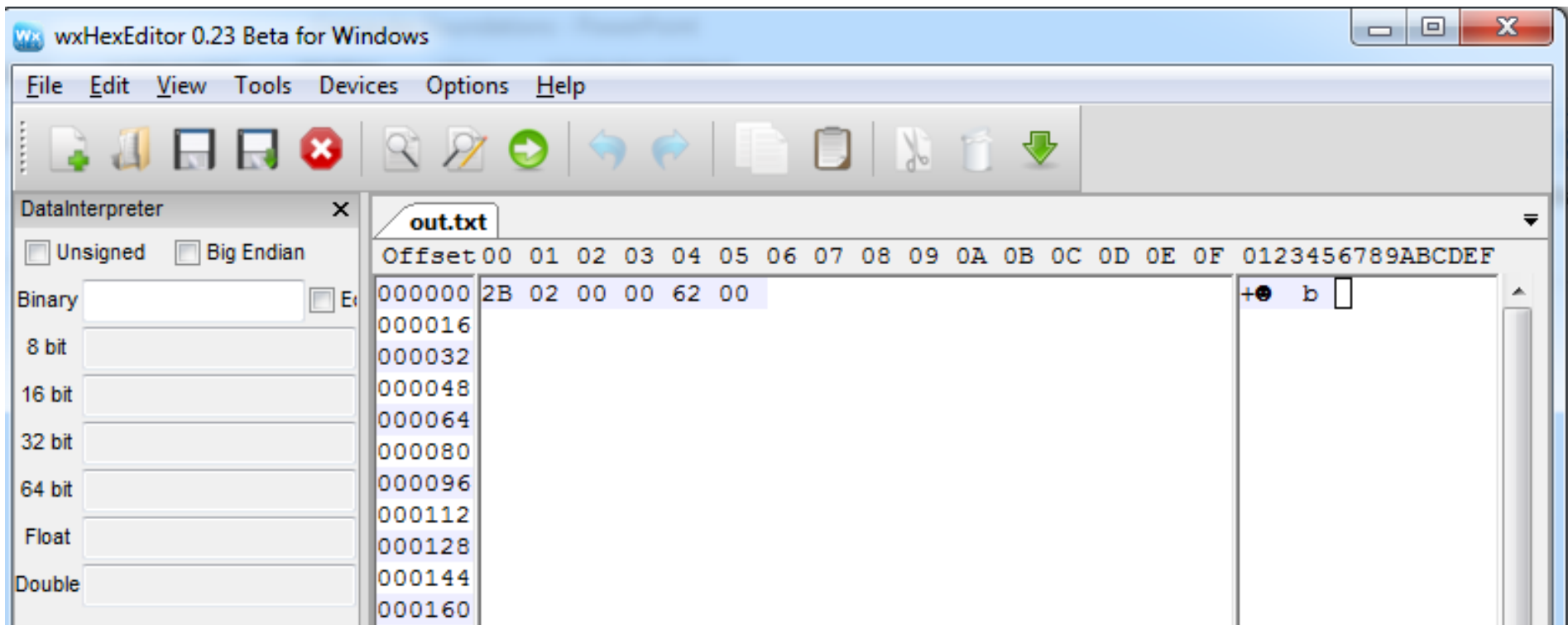
- *big-endian* architecture
- *little-endian* architecture

# Data Representation

```
#include <iostream>
#include <fstream>
using namespace std;

int main() {
    ofstream myfile;
    myfile.open("out.txt", ofstream::binary);
    int num1 = 555;
    short num2 = 98;
    myfile.write((const char*)&num1, sizeof(int));
    myfile.write((const char*)&num2, sizeof(short));
    myfile.close();
    return 0;
}
```

# wxHexEditor





# Data Representation



- How do we represent the following hexadecimal number in memory ?

0x12345678

# Data Representation



- How do we represent a string in memory, for example "5 Market St."?

# Data Representation



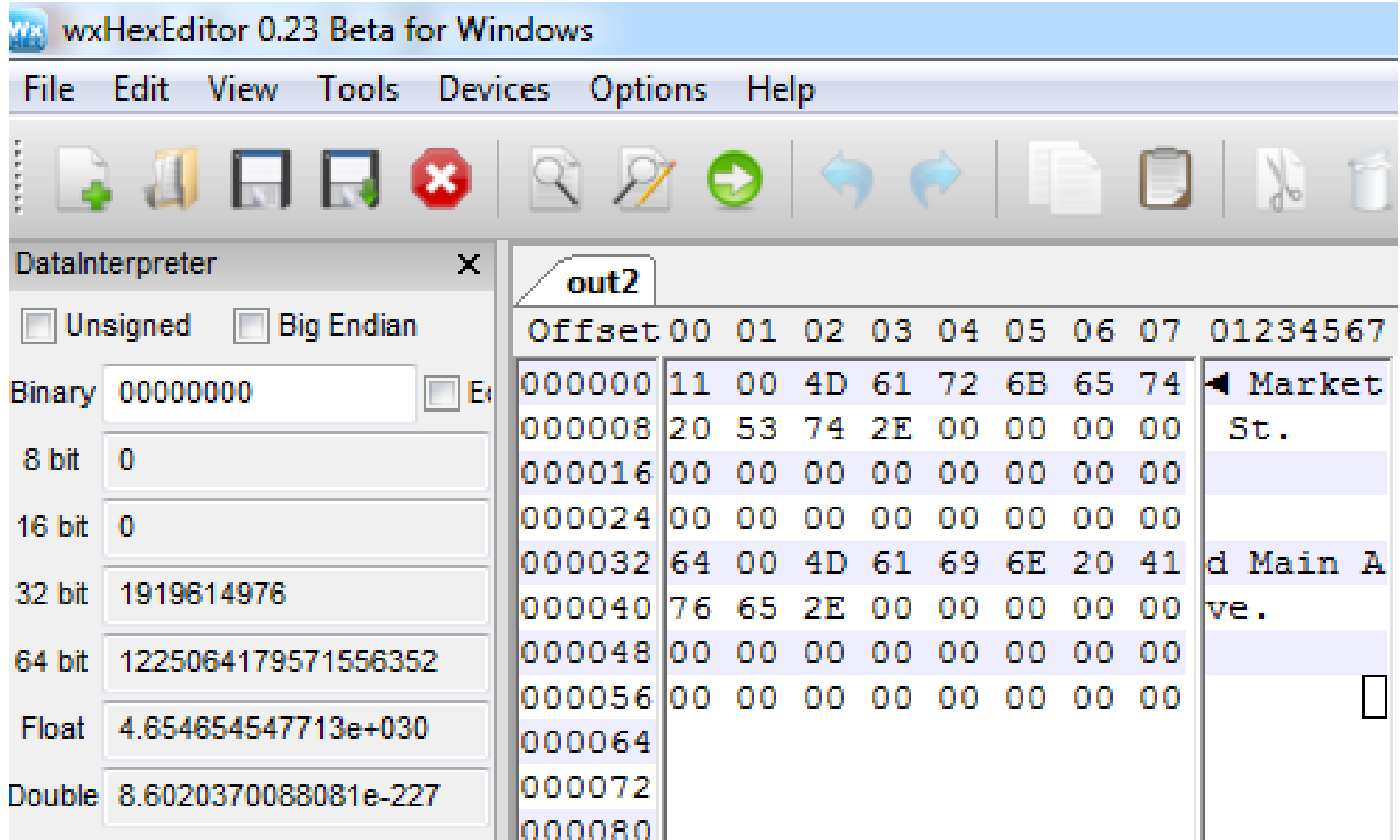
- Data Structures:
  - ▣ Are broken up into fields
  - ▣ A field has a size and name

# Data Structures

- How do we represent the following data structure:

Bytes	Description
2	street number
30	street name

# Data Structures



# Bit Flags

Bytes	Description
30	Name
1	busy
1	tired
1	hungry
1	studying
1	sleepy
1	Location included
17	Location (optional)

# Bit Flags

- How do we represent the following data structure:

Bytes	Description
30	Name
1	flags
17	Location (optional)

# Bit Flags

```
#include <iostream>
using namespace std;
int main()
{
    enum Options {BUSY = 0x01,    // 0000 0001
                  Tired = 0x02,   // 0000 0010
                  HUNGRY = 0x04,  // 0000 0100
                  STUDY = 0x08,   // 0000 1000
                  SLEEPY = 0x10,  // 0001 0000
                  LOC = 0x20, };  // 0010 0000

    // In C++11, we can use "uint8_t" instead of "unsigned char"
    unsigned char me = 0; // sleeping state
    me |= HUNGRY | SLEEPY | STUDY;
    me ^= HUNGRY;        // no longer hungry

    cout << "Am I Busy? " << (bool)(me & BUSY) << endl;
    cout << "Am I Sleepy? " << (bool)(me & SLEEPY) << endl;
    cout << "Am I Hungry? " << (bool)(me & HUNGRY) << endl;
    cout << "Am I Studying ? " << (bool)(me & STUDY) << endl;
    return 0;
}
```



# Bit Flags

wxHexEditor 0.23 Beta for Windows

File Edit View Tools Devices Options Help

Icons: New, Open, Save, Print, Close, Find, Replace, Go, Undo, Redo, Copy, Paste, Cut, Delete, Download

DataInterpreter

☐ Unsigned ☐ Big Endian

Binary: 01001010 ☐ Endian

8 bit: 74

16 bit: 24906

32 bit: 1701732682

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
000000	4A	61	6E	65	20	4D	2E	00	00	00	00	00	00	00	00	00	Jane M.
000016	00	00	00	00	00	00	00	00	00	00	00	00	00	00	33	4C	3L
000032	61	62	20	31	00	00	00	00	00	00	00	00	00	00	00	00	ab 1
000048	4A	69	6C	6C	20	47	2E	00	00	00	00	00	00	00	00	00	Jill G.
000064	00	00	00	00	00	00	00	00	00	00	00	00	00	00	05		♣



# Boot Process

# Boot Process



1. CPU Reset
2. POST process
3. Disk boot

# BIOS



- Firmware
- A collection of programs

# CPU Reset



**From:** Digital Evidence and Computer Crime 3<sup>rd</sup> edition, Eoghan Casey

# POST



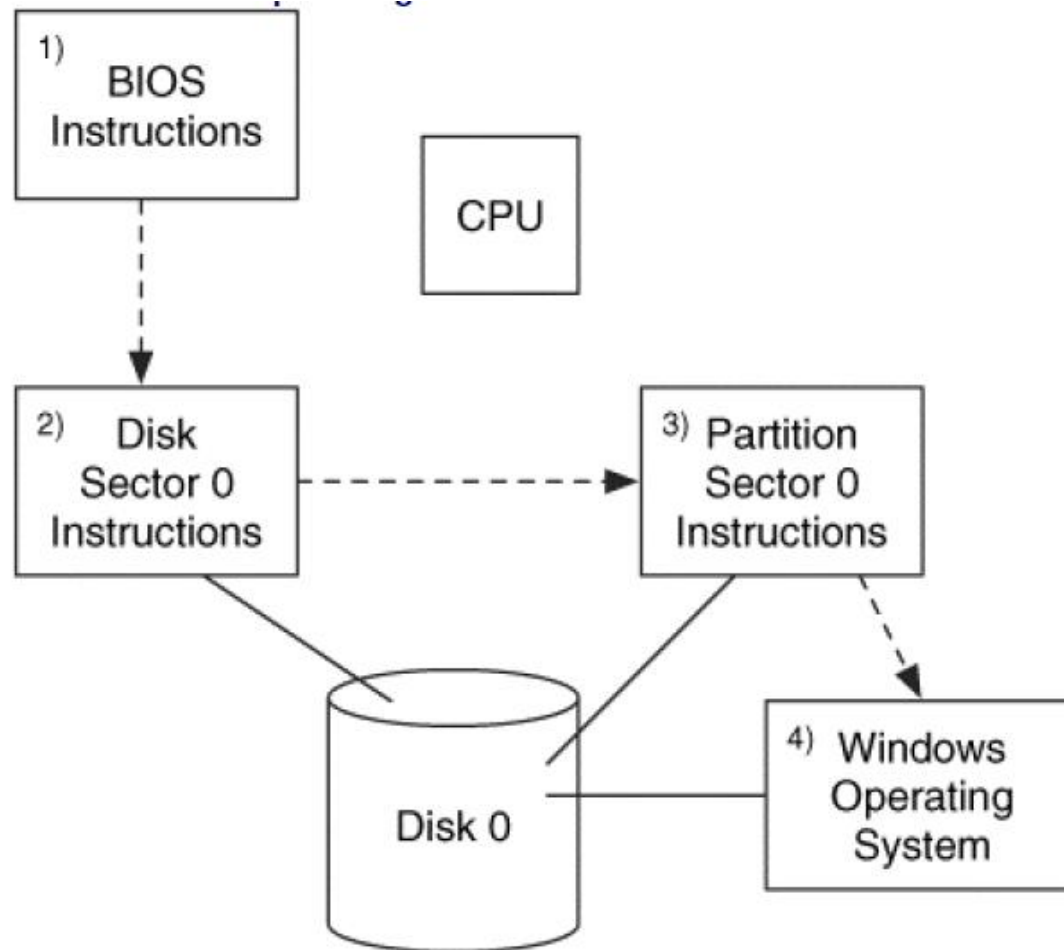
- Verification
- Hardware test
- Error reporting

# Disk Boot



1. Boot sequence governed by BIOS
2. Control passes to the MBR
3. MBR points to boot record
4. Operating system gets loaded

# Disk Boot





# Disk Boot - MBR

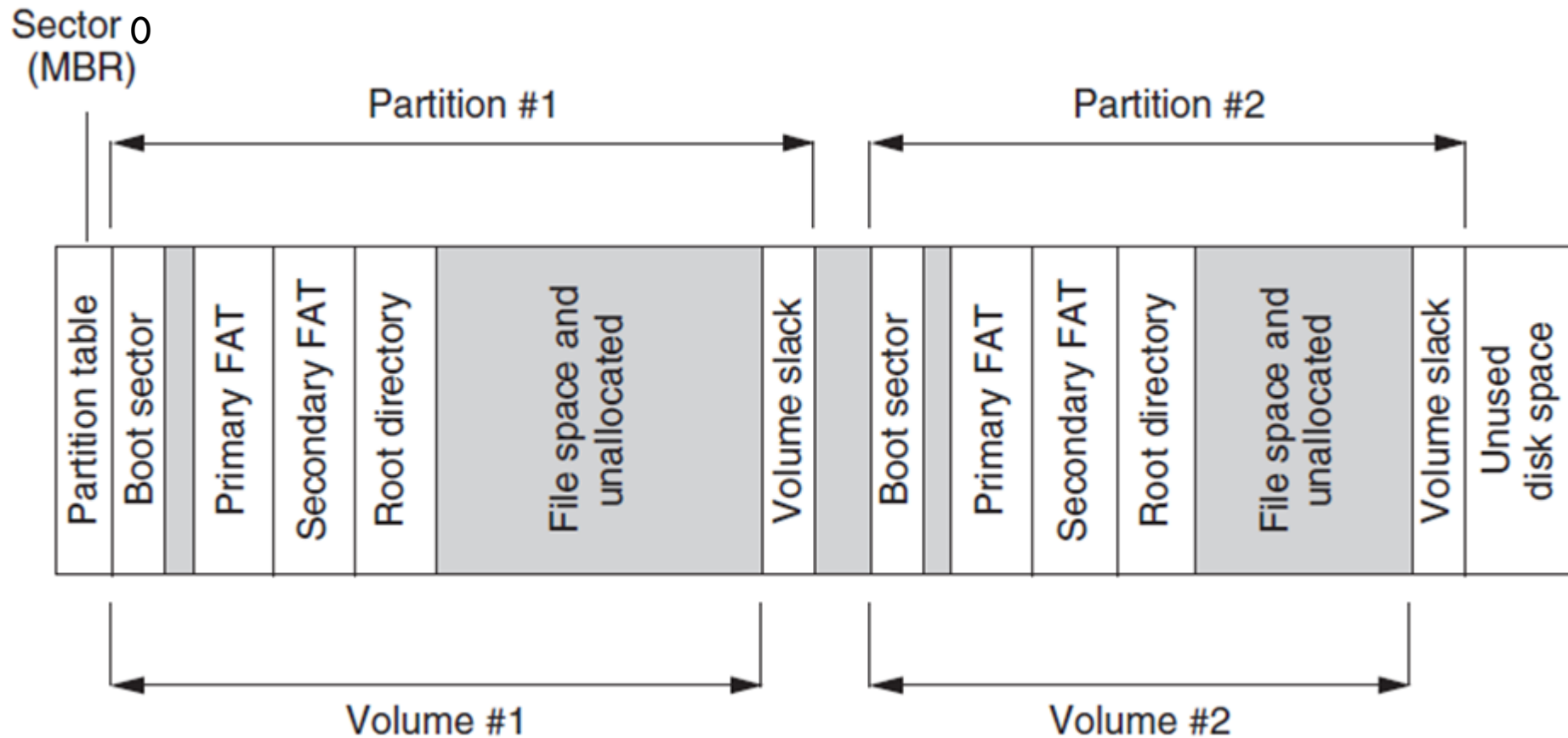
Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	33	C0	8E	D0	BC	00	7C	FB	50	07	50	1F	FC	BE	1B	7C	3ÀŽĐ4.  ûP.P.ü% .
00000010	BF	1B	06	50	57	B9	E5	01	F3	A4	CB	BD	BE	07	B1	04	¿...PW².â.ó×Ě>¼.±.
00000020	38	6E	00	7C	09	75	13	83	C5	10	E2	F4	CD	18	8B	F5	8n. .u.fĀ.âôĬ.<ô
00000030	83	C6	10	49	74	19	38	2C	74	F6	A0	B5	07	B4	07	8B	fÆ.It.8,tô u.´.<
00000040	F0	AC	3C	00	74	FC	BB	07	00	B4	0E	CD	10	EB	F2	88	ð-<.tü»...´.Ĭ.ëð^
00000050	4E	10	E8	46	00	73	2A	FE	46	10	80	7E	04	0B	74	0B	N.èF.s*pF.€~...t.
00000060	80	7E	04	0C	74	05	A0	B6	07	75	D2	80	46	02	06	83	€~...t. q.uòĚF..f
00000070	46	08	06	83	56	0A	00	E8	21	00	73	05	A0	B6	07	EB	F..fV..è!.s. q.ë
00000080	BC	81	3E	FE	7D	55	AA	74	0B	80	7E	10	00	74	C8	A0	4.>p}U*t.€~...tÈ
00000090	B7	07	EB	A9	8B	FC	1E	57	8B	F5	CB	BF	05	00	8A	56	..ë<ü.W<ôĚ¿...ŠV
000000A0	00	B4	08	CD	13	72	23	8A	C1	24	3F	98	8A	DE	8A	FC	.´.Ĭ.r#ŠĀ\$?~ŠPŠü
000000B0	43	F7	E3	8B	D1	86	D6	B1	06	D2	EE	42	F7	E2	39	56	C÷â<Ň+Ô±.ÒĬB÷â9V
000000C0	0A	77	23	72	05	39	46	08	73	1C	B8	01	02	BB	00	7C	.w#r.9F.s.,...».
000000D0	8B	4E	02	8B	56	00	CD	13	73	51	4F	74	4E	32	E4	8A	<N.<V.Ĭ.sQOtN2âŠ
000000E0	56	00	CD	13	EB	E4	8A	56	00	60	BB	AA	55	B4	41	CD	V.Ĭ.ëâŠV.`»*U`AĬ
000000F0	13	72	36	81	FB	55	AA	75	30	F6	C1	01	74	2B	61	60	.r6.ûU*uoôĀ.t+a`
00000100	6A	00	6A	00	FF	76	0A	FF	76	08	6A	00	68	00	7C	6A	j.j.ÿv.ÿv.j.h. j
00000110	01	6A	10	B4	42	8B	F4	CD	13	61	61	73	0E	4F	74	0B	.j.´B<ôĬ.aas.Ot.
00000120	32	E4	8A	56	00	CD	13	EB	D6	61	F9	C3	49	6E	76	61	2âŠV.Ĭ.ëÔaûĀInva
00000130	6C	69	64	20	70	61	72	74	69	74	69	6F	6E	20	74	61	lid partition ta
00000140	62	6C	65	00	45	72	72	6F	72	20	6C	6F	61	64	69	6E	ble.Error loadin
00000150	67	20	6F	70	65	72	61	74	69	6E	67	20	73	79	73	74	g operating syst
00000160	65	6D	00	4D	69	73	73	69	6E	67	20	6F	70	65	72	61	em.Missing opera
00000170	74	69	6E	67	20	73	79	73	74	65	6D	00	00	00	00	00	ting system.....
00000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000001B0	00	00	00	00	00	2C	44	63	18	2E	07	C3	00	00	80	00	.....,Dc...Ā...Ě.
000001C0	01	01	0C	4F	D0	52	80	1F	00	00	80	AA	E6	00	00	00	...ÔĐRĚ...€²æ...
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	.....U²

# Disk Boot – Partition Table

Structure of a 16-byte Partition Table Entry		
Relative Offsets ( <i>within entry</i> )	Length (bytes)	Contents
0	1	Boot Indicator (80h = <i>active</i> )
1 - 3	3	Starting CHS values
4	1	<i>Partition-type</i> Descriptor
5 - 7	3	Ending CHS values
8 - 11	4	Starting Sector
12 - 15	4	Partition Size (in sectors)

From: <http://thestarman.pcministry.com/asm/mbr/PartTables.htm>

# Disk Boot - Disk Partition



**From:** Digital Evidence and Computer Crime 3<sup>rd</sup> edition, Eoghan Casey



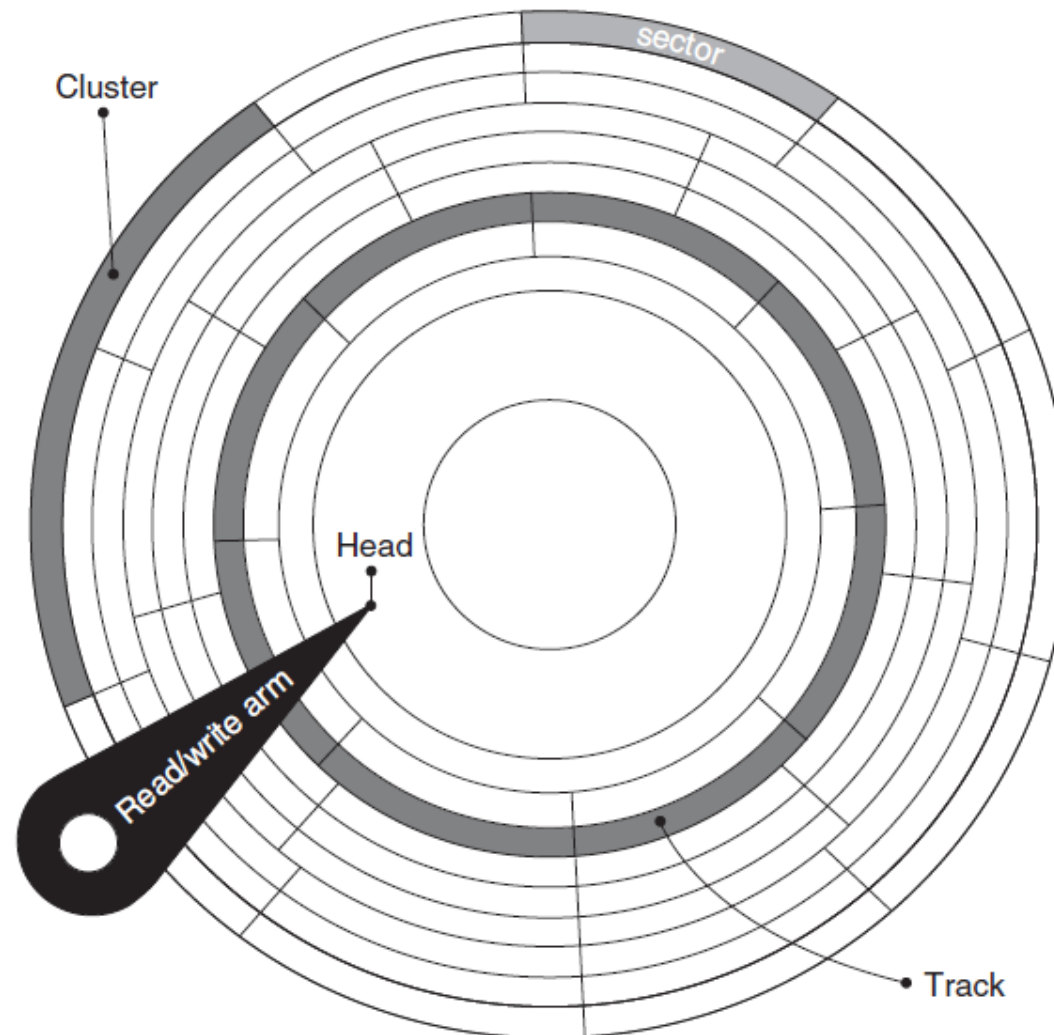
# Hard Disk

# Hard Disk



- Platters
- Track
- Cylinder
- Sector
- Cluster

# Hard Disk - CHS addressing



**From:** Digital Evidence and Computer Crime 3<sup>rd</sup> edition, Eoghan Casey

# *Logical Block Addresses*



$$\text{LBA} = (((C * \text{hpc}) + H) * \text{spt}) + S - 1$$

Where:

- hpc: heads per cylinder
- spt: sectors per track

# *Logical Block Addresses*

- Suppose we have a hard disk with:
  - ▣ 16 heads
  - ▣ 63 sectors per track
- What is the LBA for a CHS address = 2, 3, 4?



# Cluster

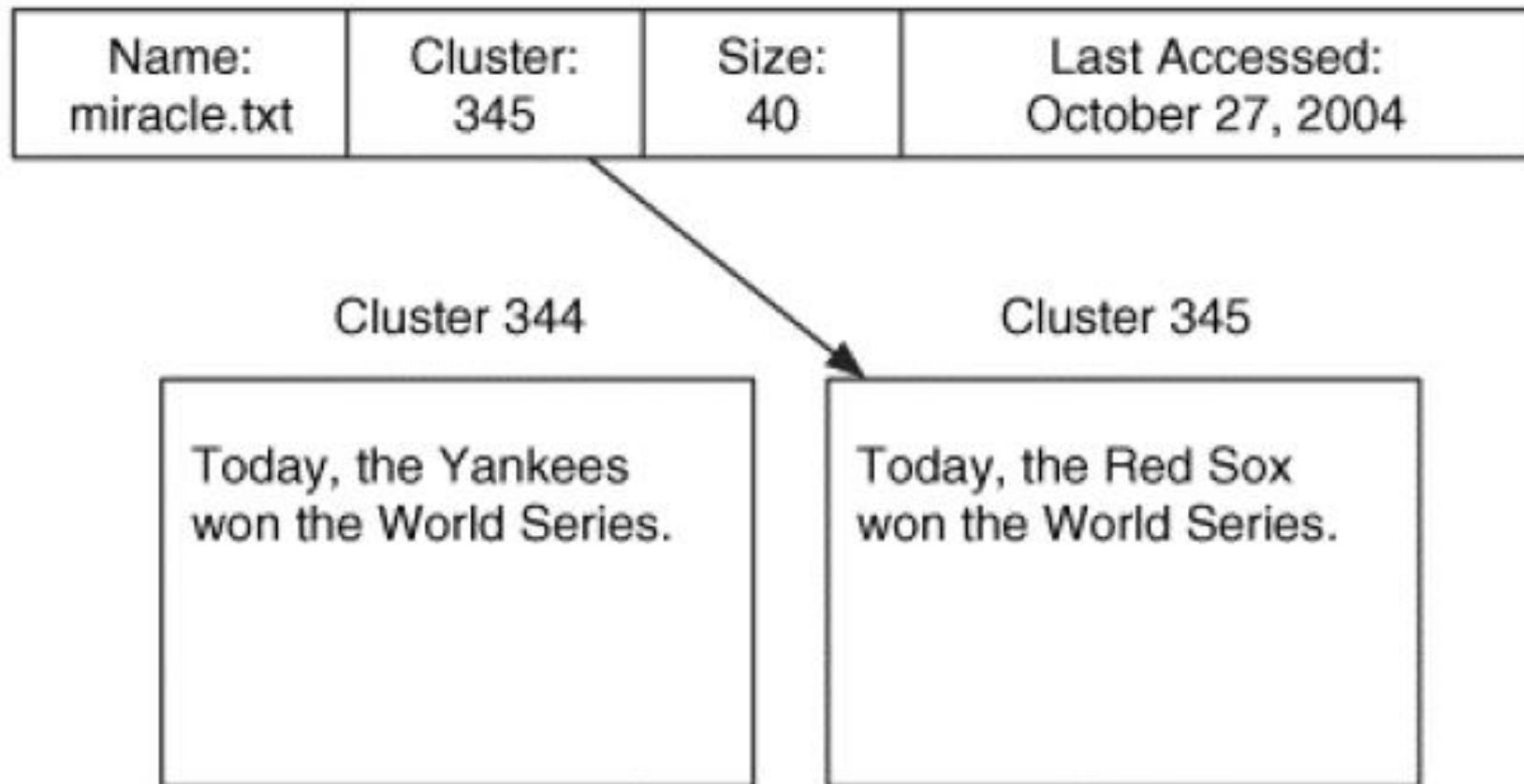
File System	Cluster Min/Max
FAT12	512B/2KB
FAT16	2KB/32KB
FAT32	2KB/32KB
NTFS 1.1	512B/8KB
NTFS 3.0	512B/64KB

# File Systems



- What is a file system?

# File Systems



# Self-Monitoring, Analysis, and Reporting Technology (smartmontools)

ID#	ATTRIBUTE_NAME	FLAG	VALUE	WORST	THRESH	TYPE	UPDATED	WHEN_FAILED	RAW_VALUE
9	Power_On_Hours	0x0032	099	099	---	old_age	Always	-	1482
12	Power_Cycle_Count	0x0032	100	100	---	old_age	Always	-	1657

# References



1. File System Forensic Analysis, 2<sup>nd</sup> edition, Brian Carrier, 2005.
2. Digital Evidence and Computer Crime, 3<sup>rd</sup> edition, Eoghan Casey, 2011.
3. Digital Archaeology: The Art and Science of Digital Forensics, Michael W Graves, 2013