Emulating a Game Boy

In .NET 6

Code!

https://github.com/wcabus/gb-net6

Reference sessions

- The Ultimate Game Boy Talk by Michael Steil
  - https://www.youtube.com/watch?v=HyzD8pNlpwI
- Building a Gameboy Emulator by David Whitney
  - https://www.youtube.com/watch?v=pP4lumdqpXY

Hardware information

- Technical documentation by the community
  - https://gbdev.io/pandocs/
- SM83 opcodes
  - https://gbdev.io/gb-opcodes/optables/
- Architecture of the Game Boy:
  - https://www.copetti.org/writings/consoles/game-boy/

Emulation

- Debug emulator:
  - https://bgb.bircd.org/#downloads
- Game Boy cartridge reader/writer, flash carts (not sponsored):
  - https://shop.insidegadgets.com/

Nintendo®

Hello there

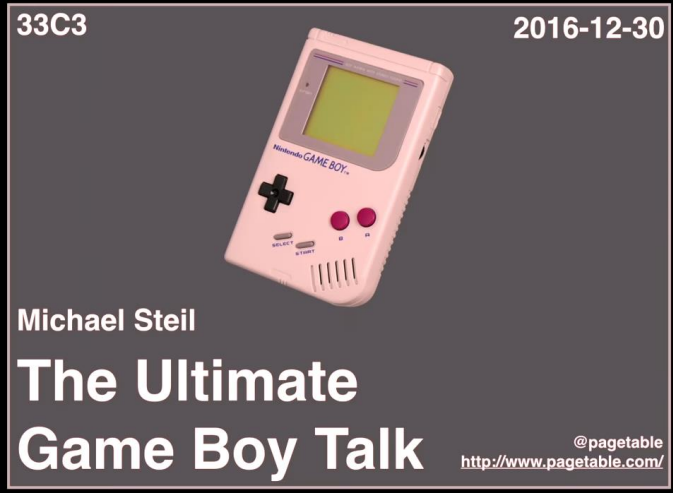
I’m a Coding Architect at Xpirit.

I try not to spend too much money on retro gaming consoles.

I love tinkering with technology and finding out how stuff works.

Credits

Session by David Whitney



Session by Michael Steil

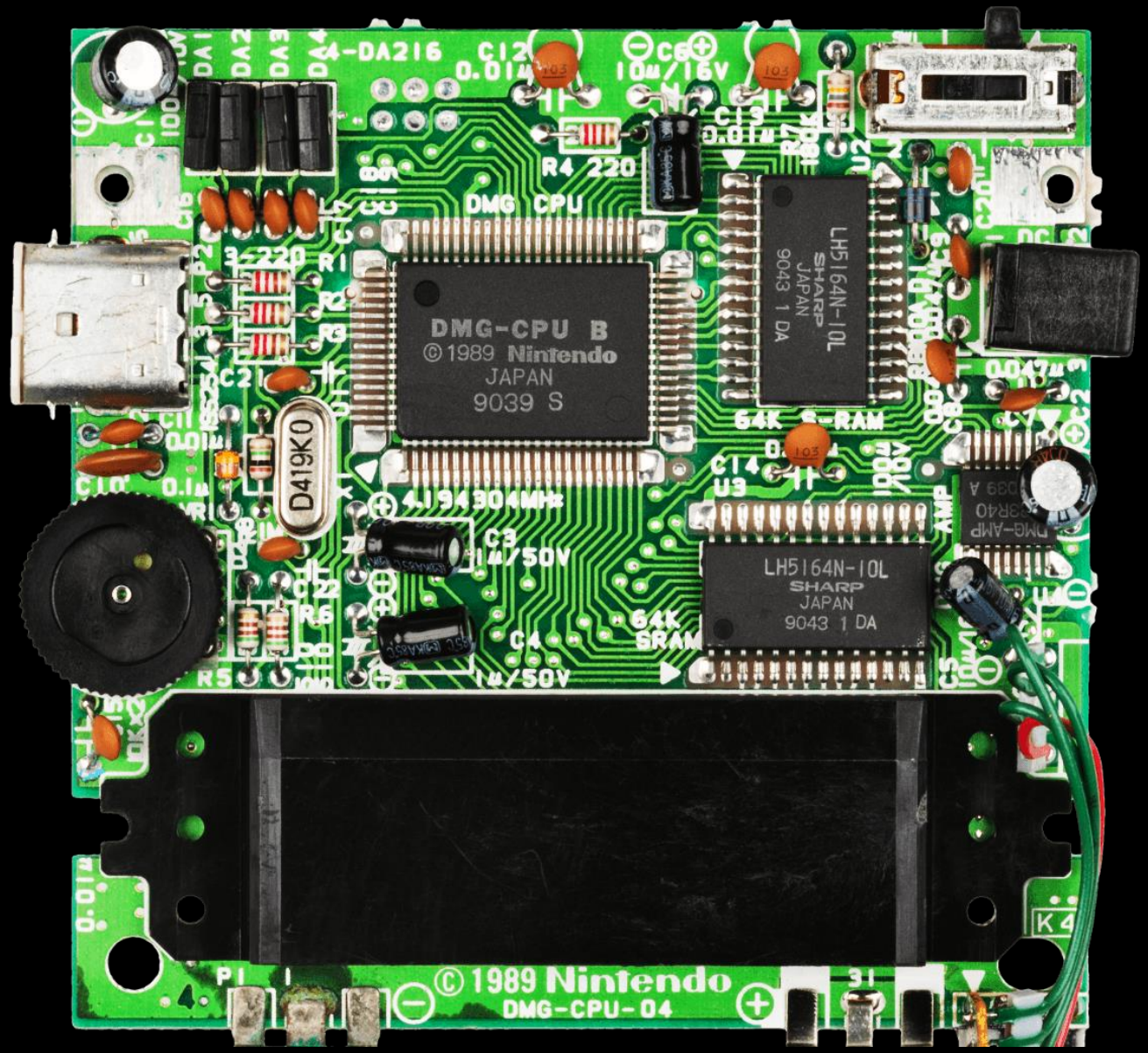
Ooh, that sounds like a fun project to ~~waste~~ spend my free time on!

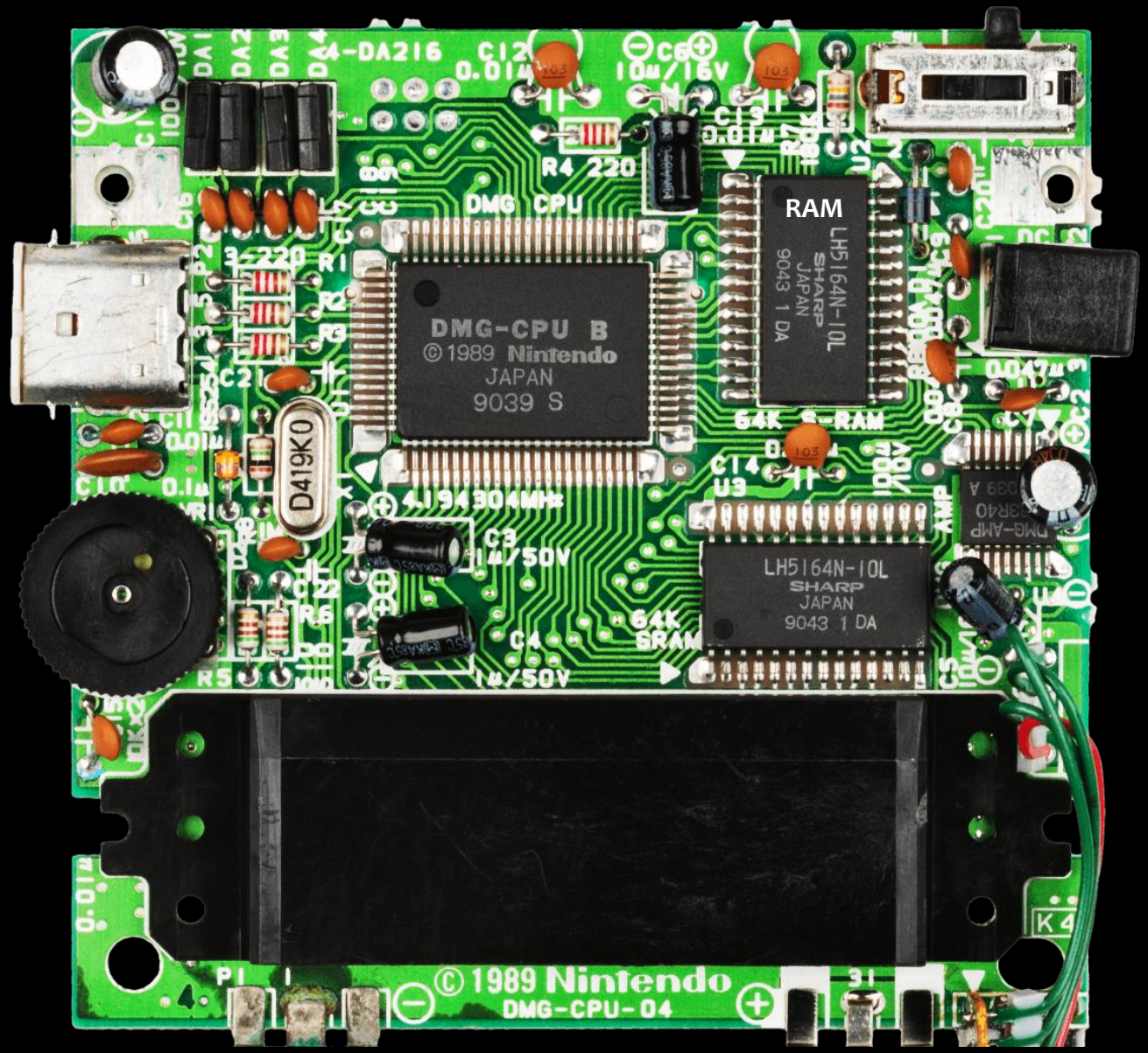
The hardware specs

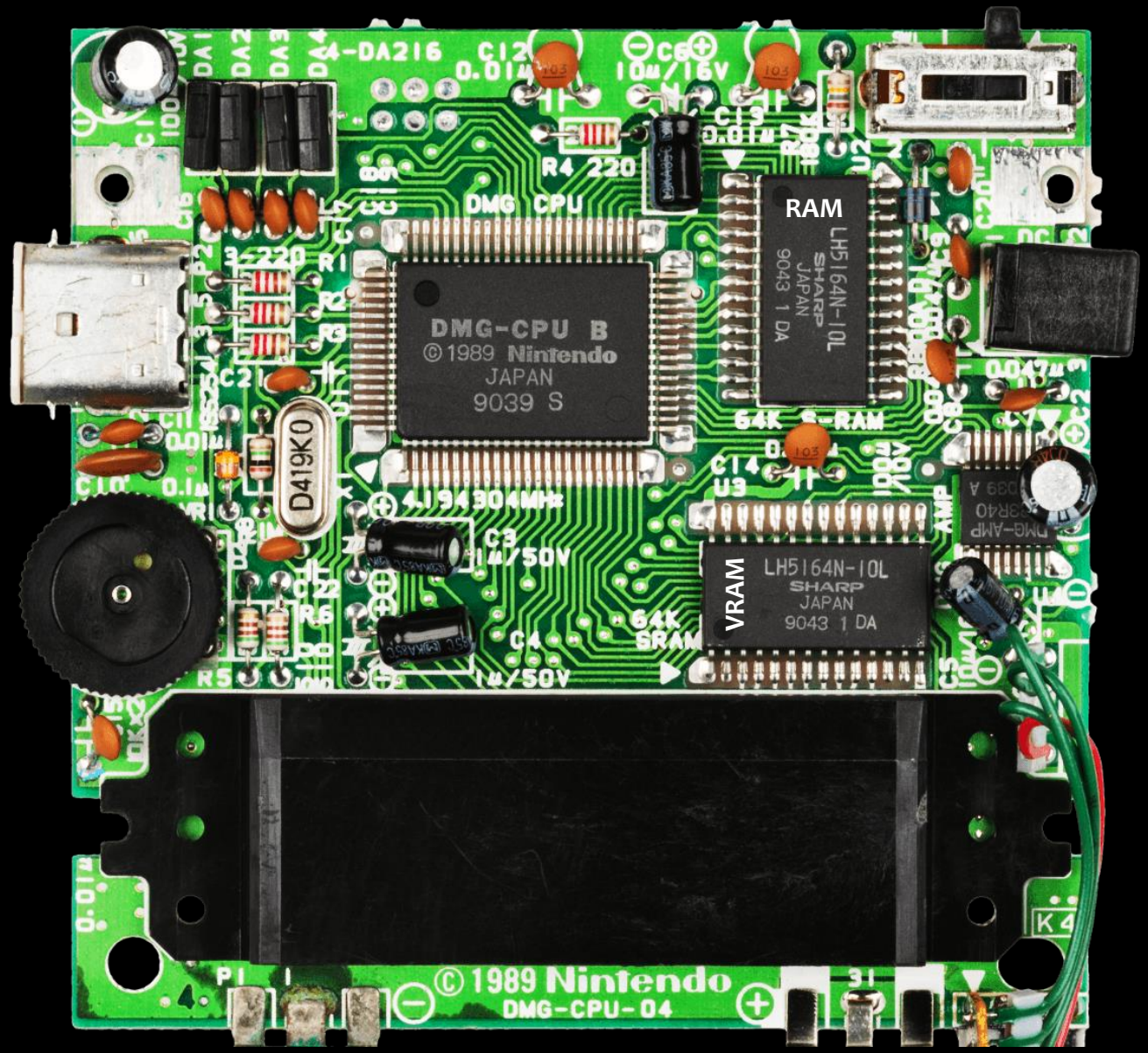
Of the DMG-01

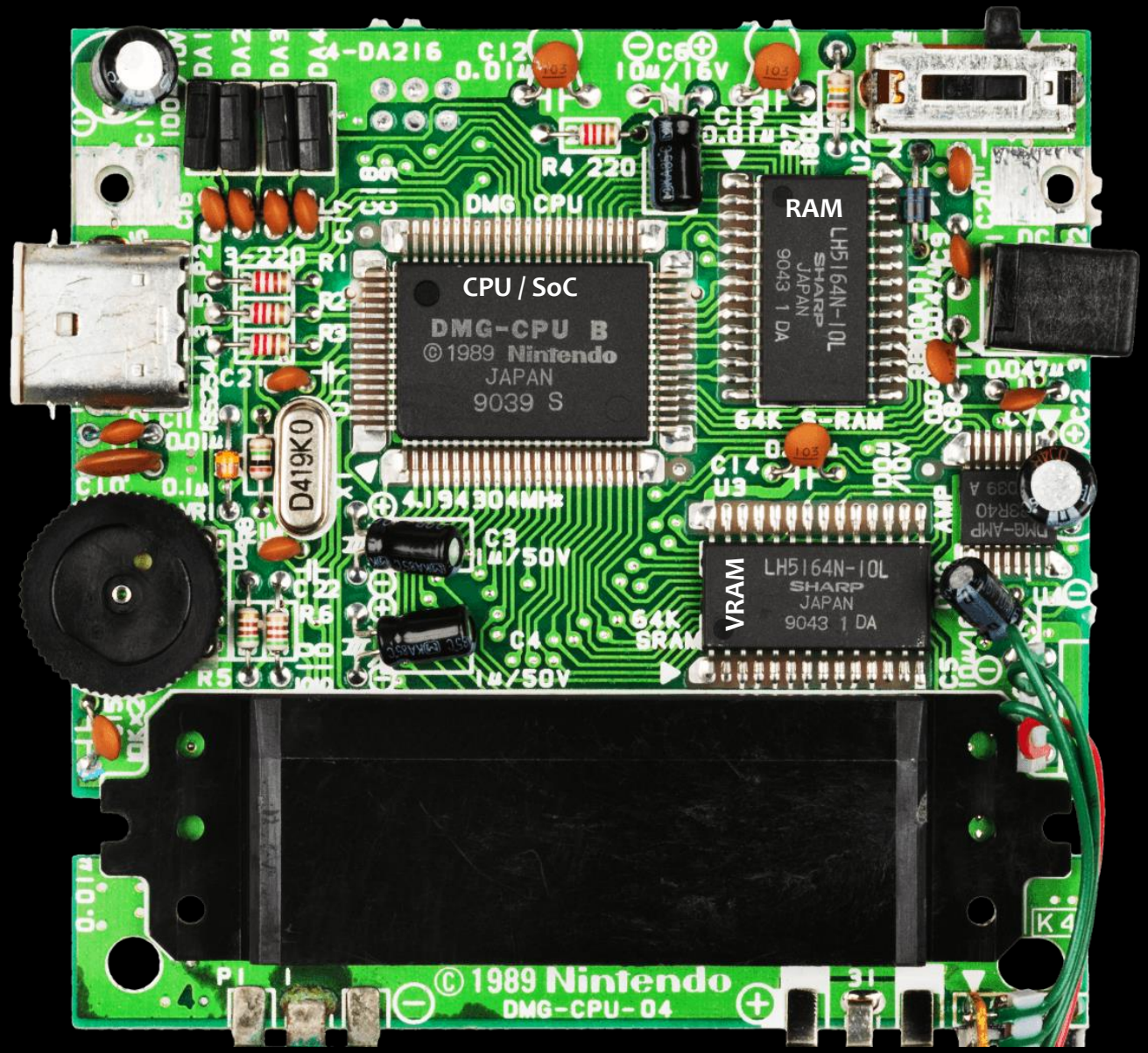
The hardware specs

- CPU: 8-bit Sharp SM83
  4.19 MHz (but not really)
- 8 KiB work RAM
- 8 KiB video RAM
- 256 bytes boot ROM
- 2.6” LCD with 4 “colors” and a resolution of 160 x 144 pixels
- 2 pulse wave channels
  1 noise channel
  1 wave sample channel

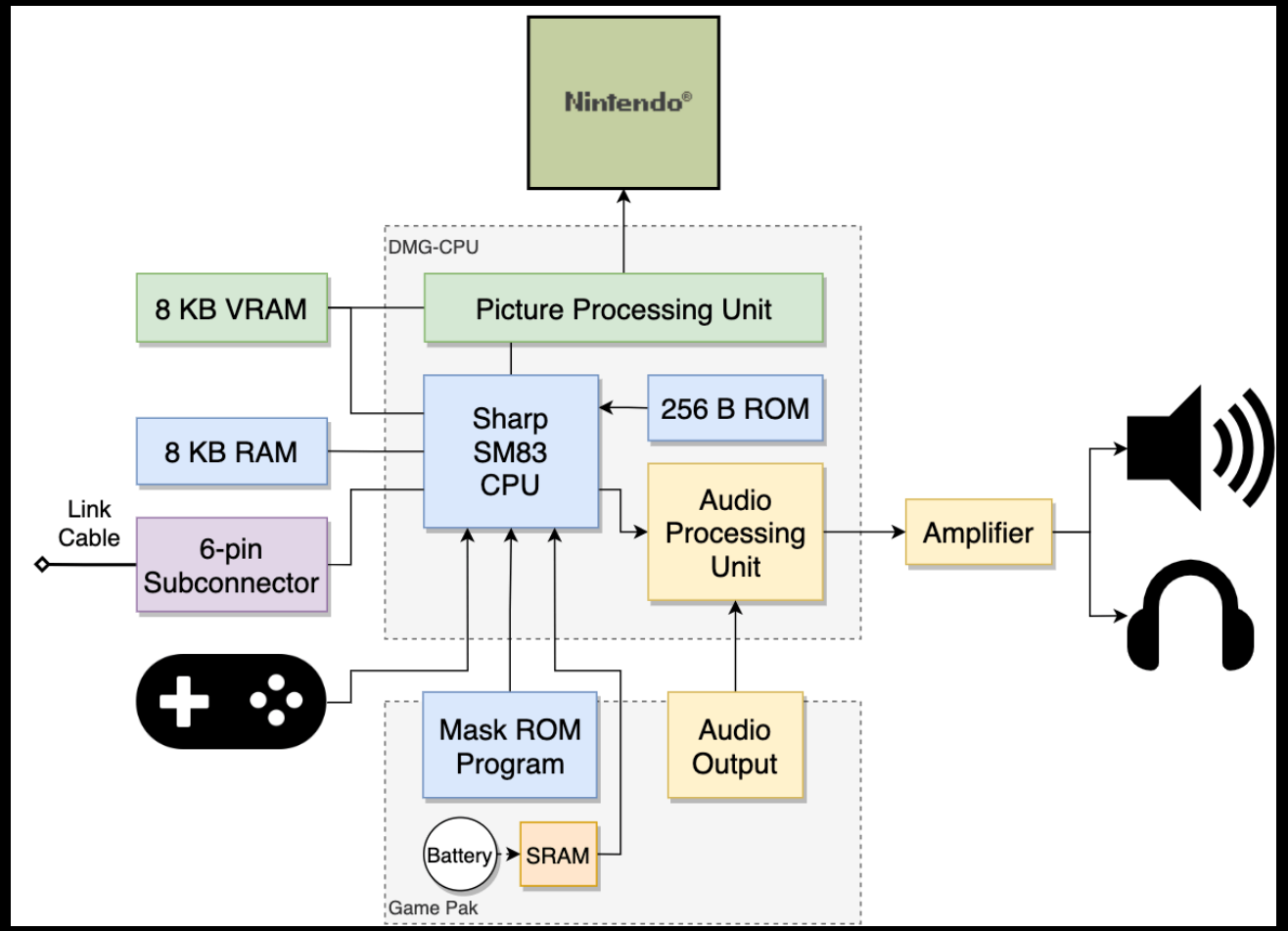

DMG CPU
DMG-CPU B
©1989 Nintendo
JAPAN
9039 S
LH5164N-10L
SHARP
JAPAN
9043 1 DA
64K S-RAM
LH5164N-10L
SHARP
JAPAN
9043 1 DA
64K SRAM
D419K0
4.194304MHz
R4 220
C6 10u/16V
C12 0.01u
C13 0.01u
C3 14/50V
C4 14/50V
DMG-AMP
©1989 Nintendo
DMG-CPU-04


RAM
DMG CPU
DMG-CPU B
©1989 Nintendo
JAPAN
9039 S
LH5164N-10L
SHARP
JAPAN
9043 1 DA
64K S-RAM
LH5164N-10L
SHARP
JAPAN
9043 1 DA
64K SRAM
D419K0
4.194304MHz
R4 220
C3 14/50V
C4 14/50V
C12 0.01μ
C6 10μ/16V
C13 0.01μ
©1989 Nintendo
DMG-CPU-04


RAM
VRAM
DMG CPU
DMG-CPU B
©1989 Nintendo
JAPAN
9039 S
LH5164N-10L
SHARP
JAPAN
9043 1 DA
D419K0
4.194304MHz
64K S-RAM
64K SRAM
©1989 Nintendo
DMG-CPU-04


CPU / SoC
DMG-CPU B
©1989 Nintendo
JAPAN
9039 S
RAM
LH5164N-10L
SHARP
JAPAN
9043 1 DA
VRAM
LH5164N-10L
SHARP
JAPAN
9043 1 DA
D419K0
4.194304MHz
©1989 Nintendo
DMG-CPU-04

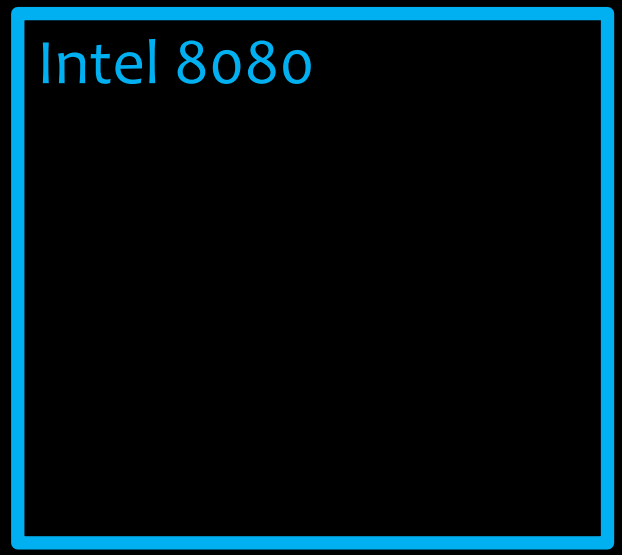
The DMG-CPU / SoC



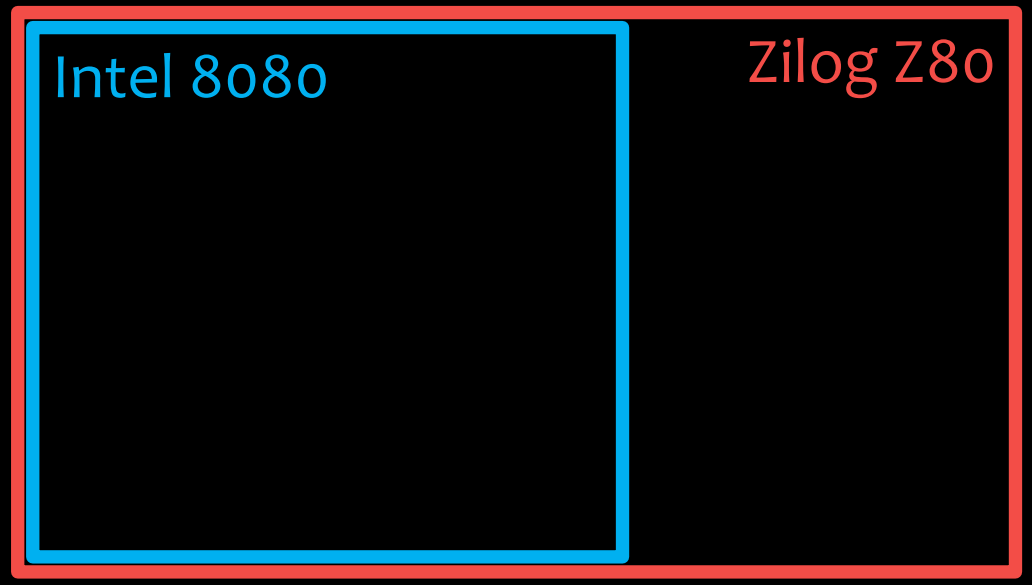
The hardware specs – Game Pak

- Different ROM sizes:
  - 32 / 64 / 128 / 256 / 512 KiB
  - 1 / 2 / 4 / 8 MiB
- Only 32 KiB can be addressed by the Game Boy
  - First 16 KiB is usually fixed
  - Second 16 KiB can be switched
- Different MBC types
  - Some support adding RAM as well, backed by a battery for save games

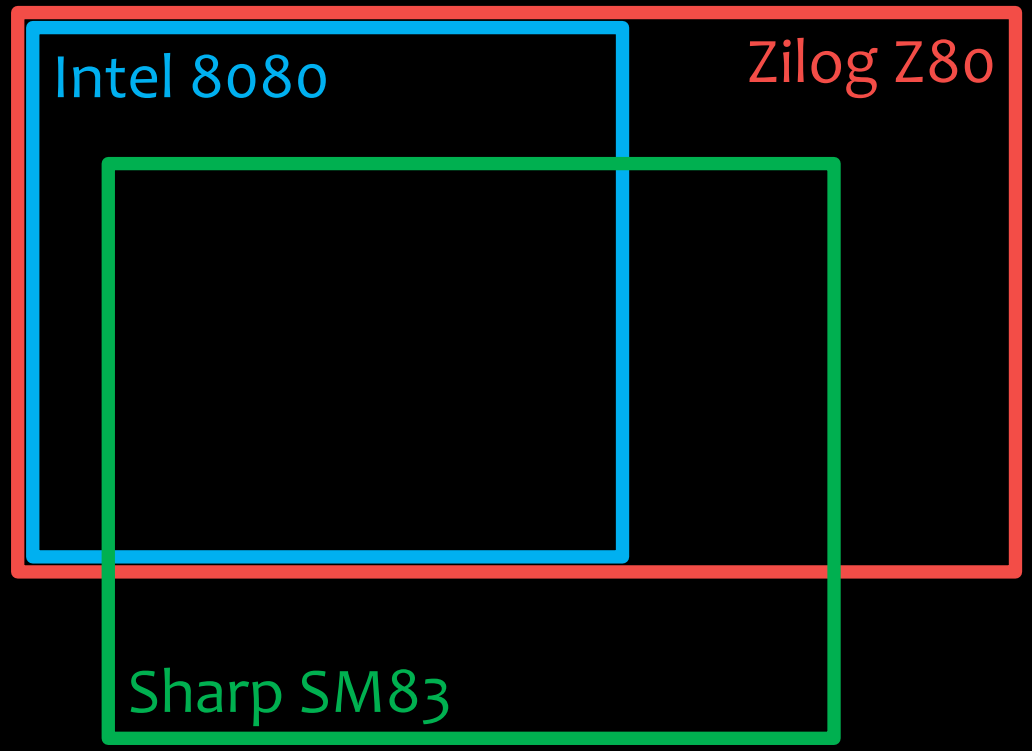
The SM83



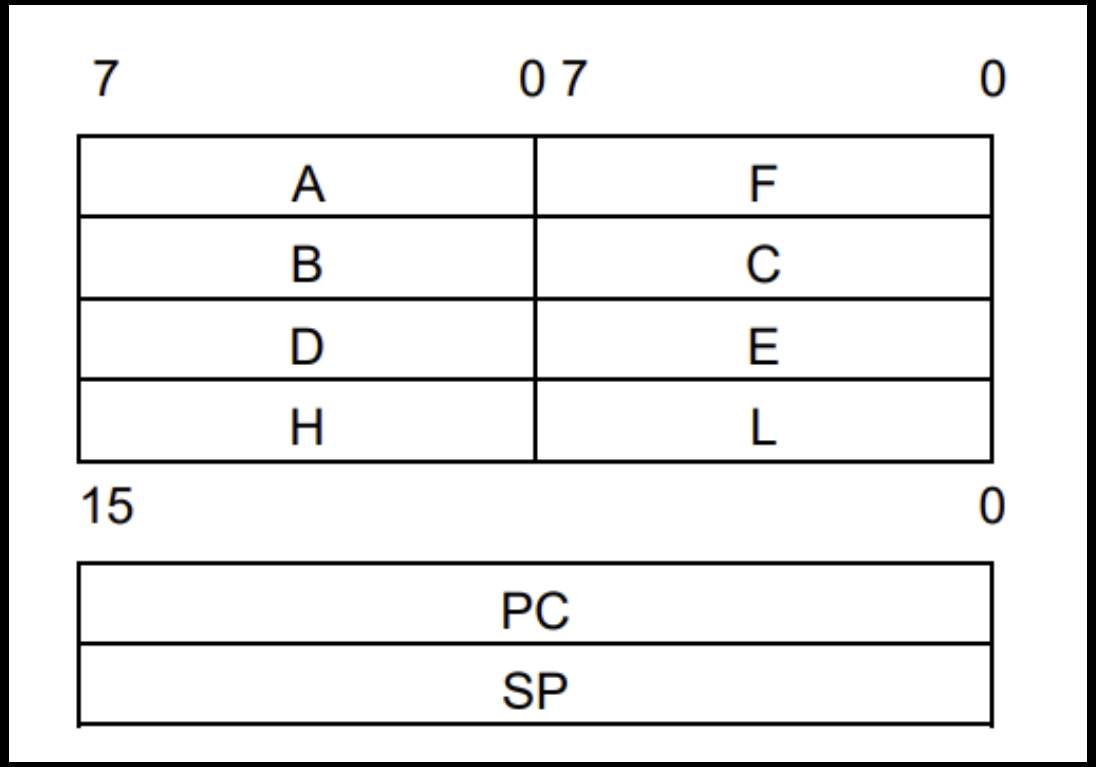
The SM83



The SM83

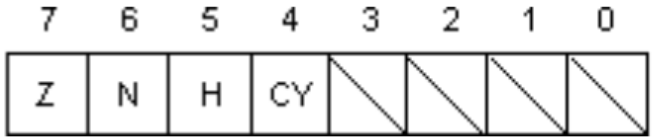


The SM83



The SM83

- Flag Register: F
  Consists of 4 flags that are set and reset according to the results of instruction execution.
  Flags CY and Z are tested by various conditional branch instructions.



Z: Set to 1 when the result of an operation is 0; otherwise reset.
N: Set to 1 following execution of the substruction instruction, regardless of the result.
H: Set to 1 when an operation results in carrying from or borrowing to bit 3.
CY: Set to 1 when an operation results in carrying from or borrowing to bit 7.

The PPU – Picture Processing Unit

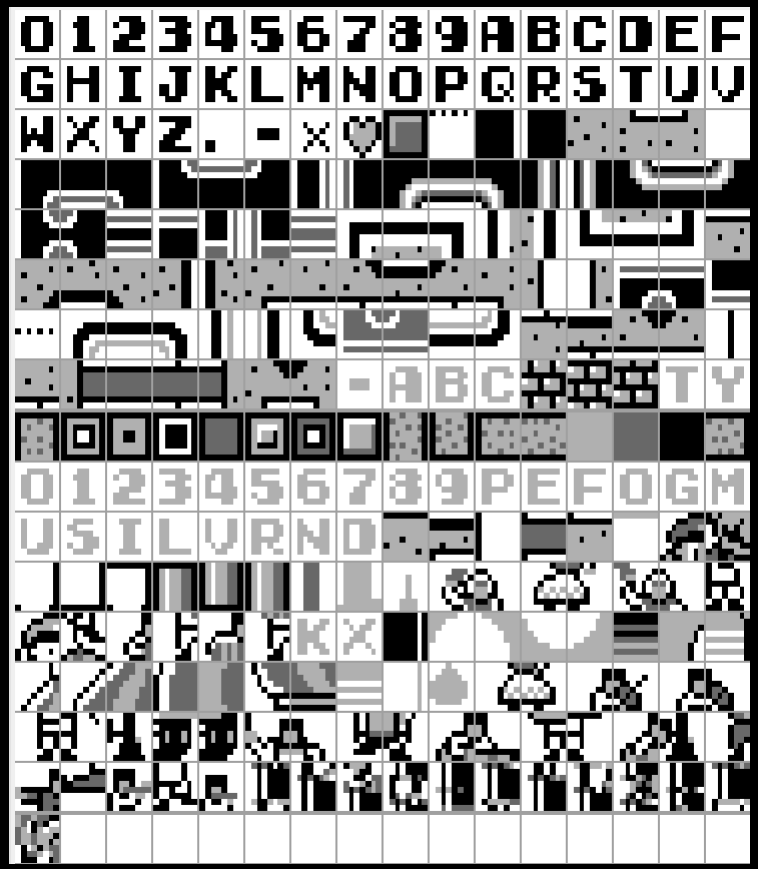
The PPU – Picture Processing Unit

#9BBC0F

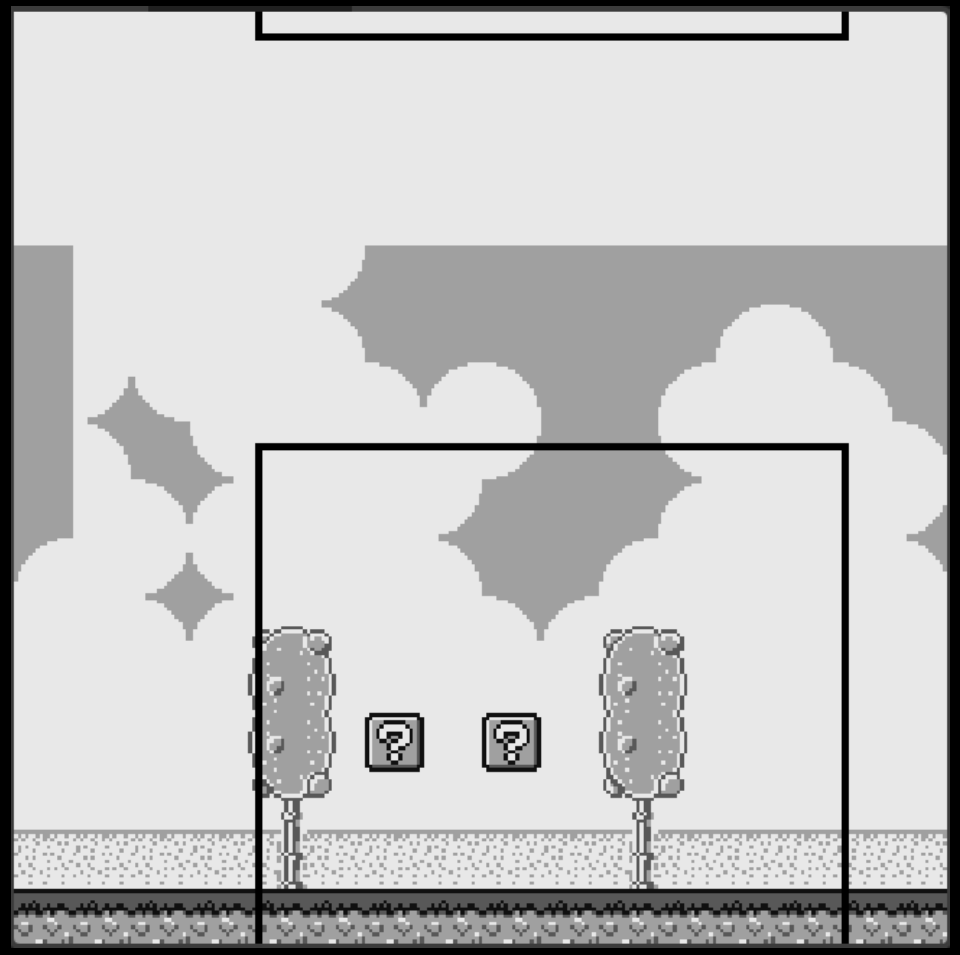
#8BAC0F

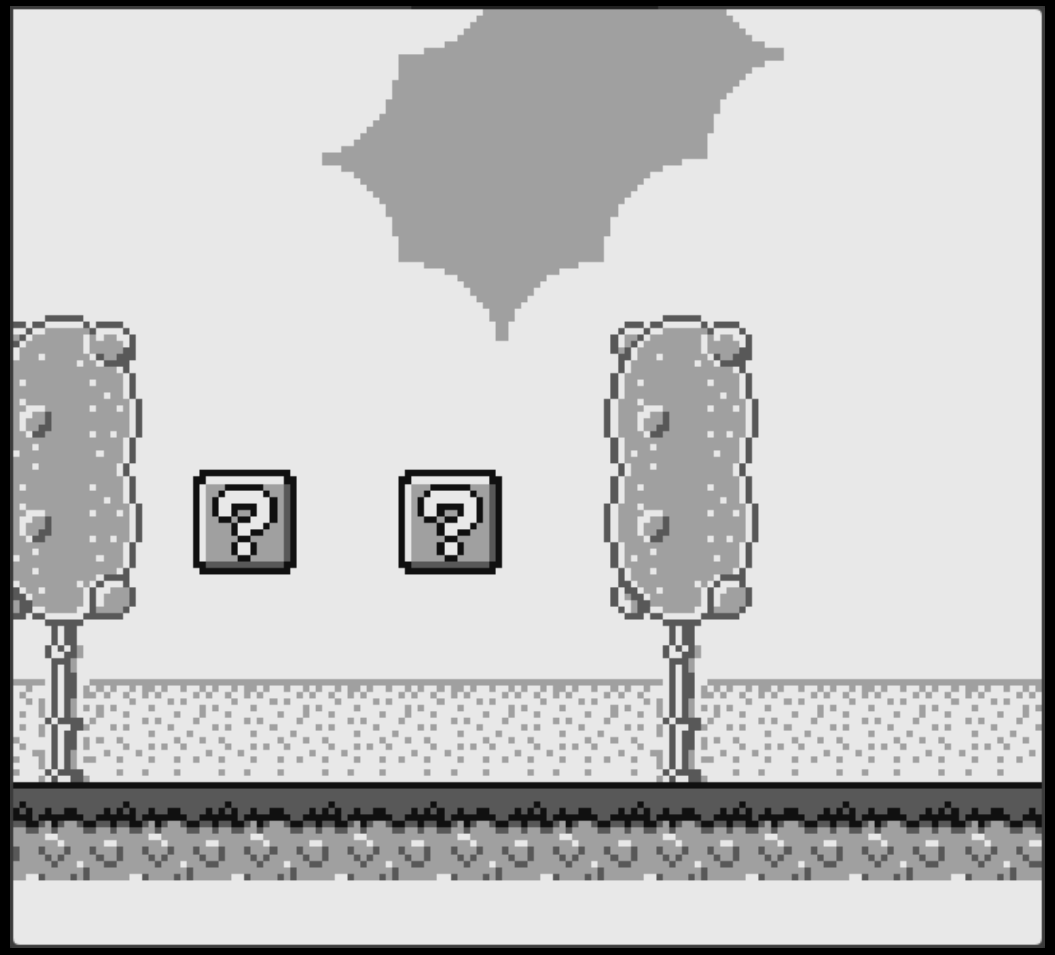
#306230

#0F380F

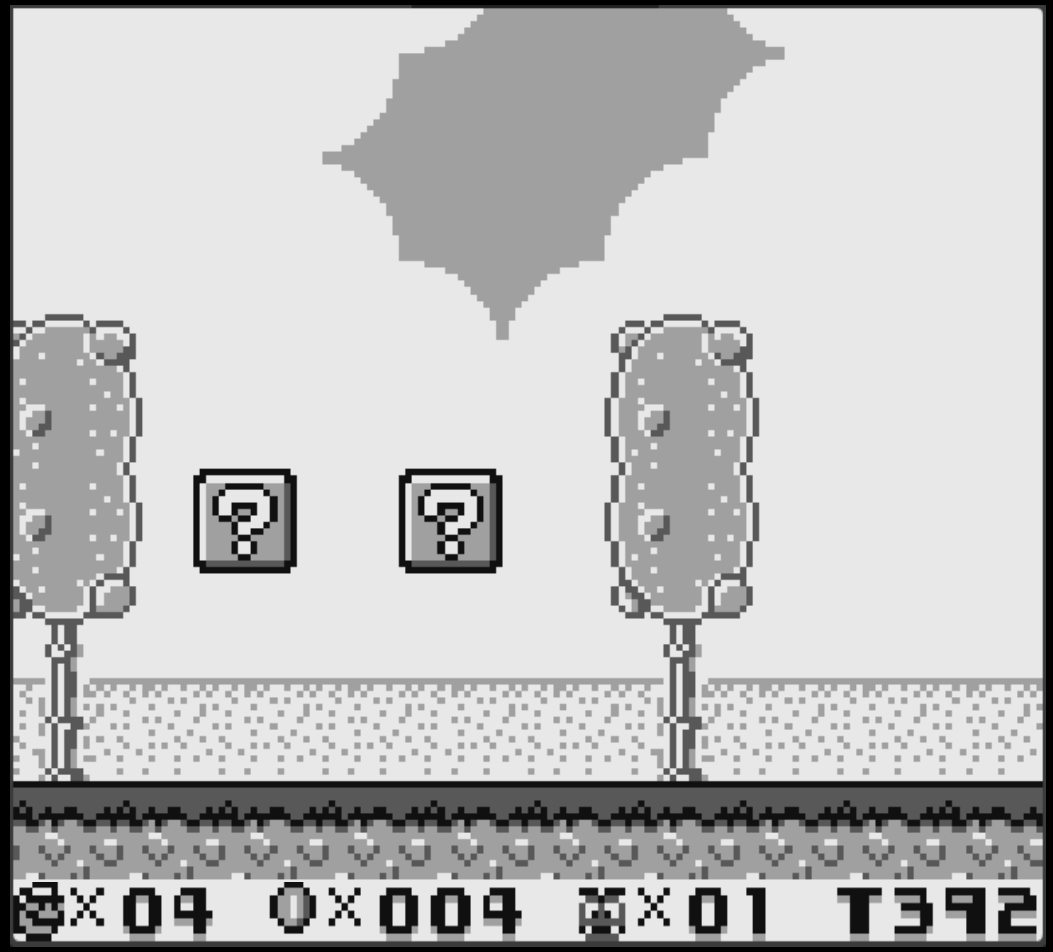


The PPU – Picture Processing Unit

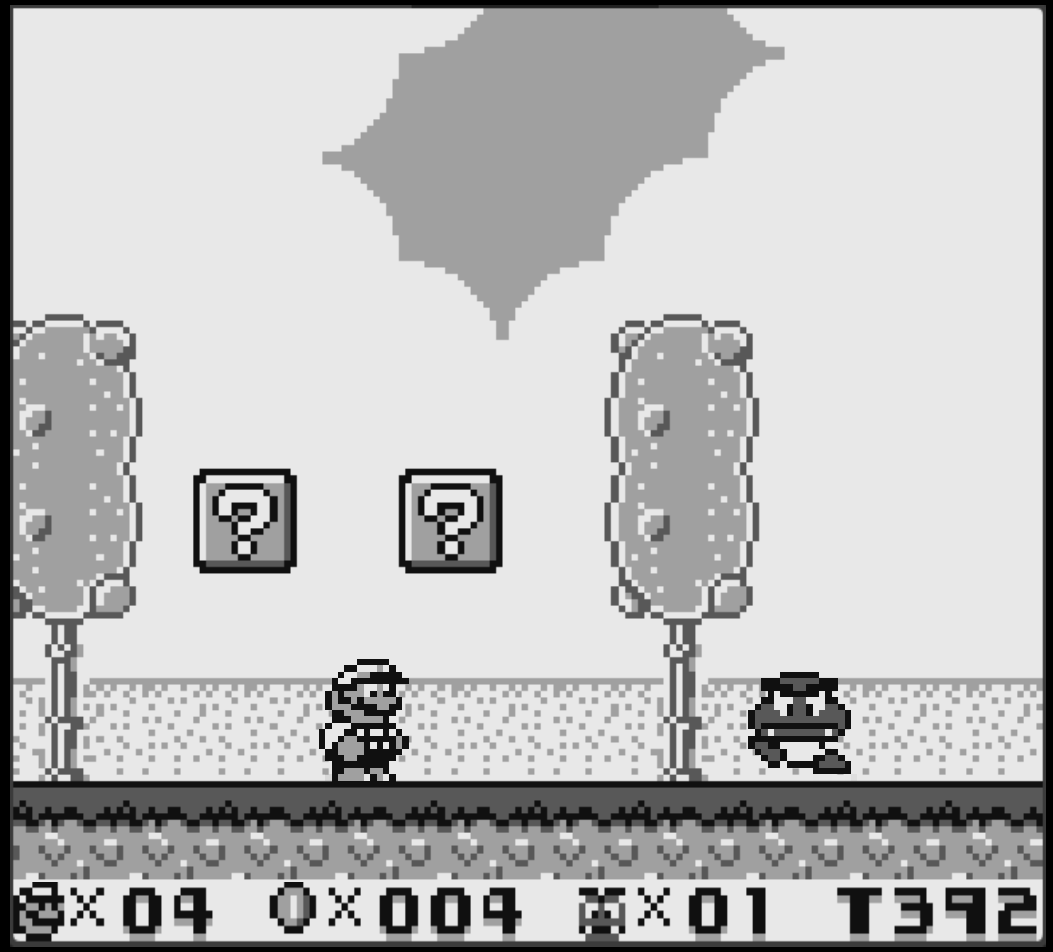




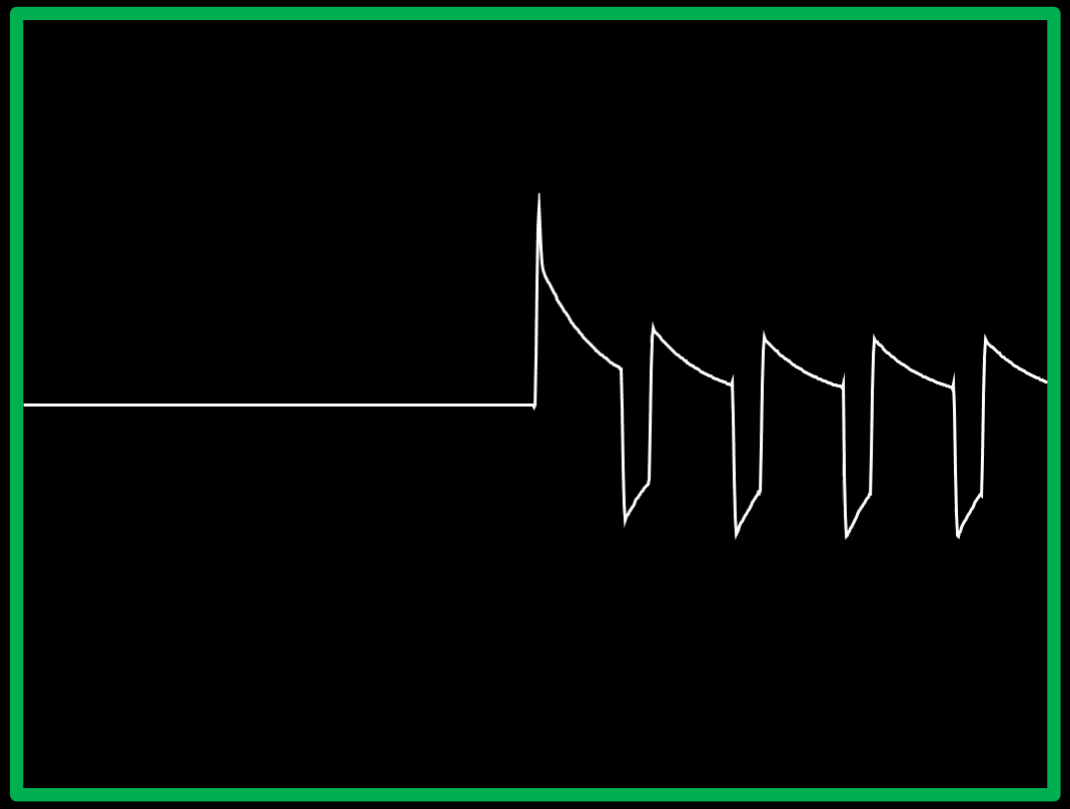
The PPU – Picture Processing Unit

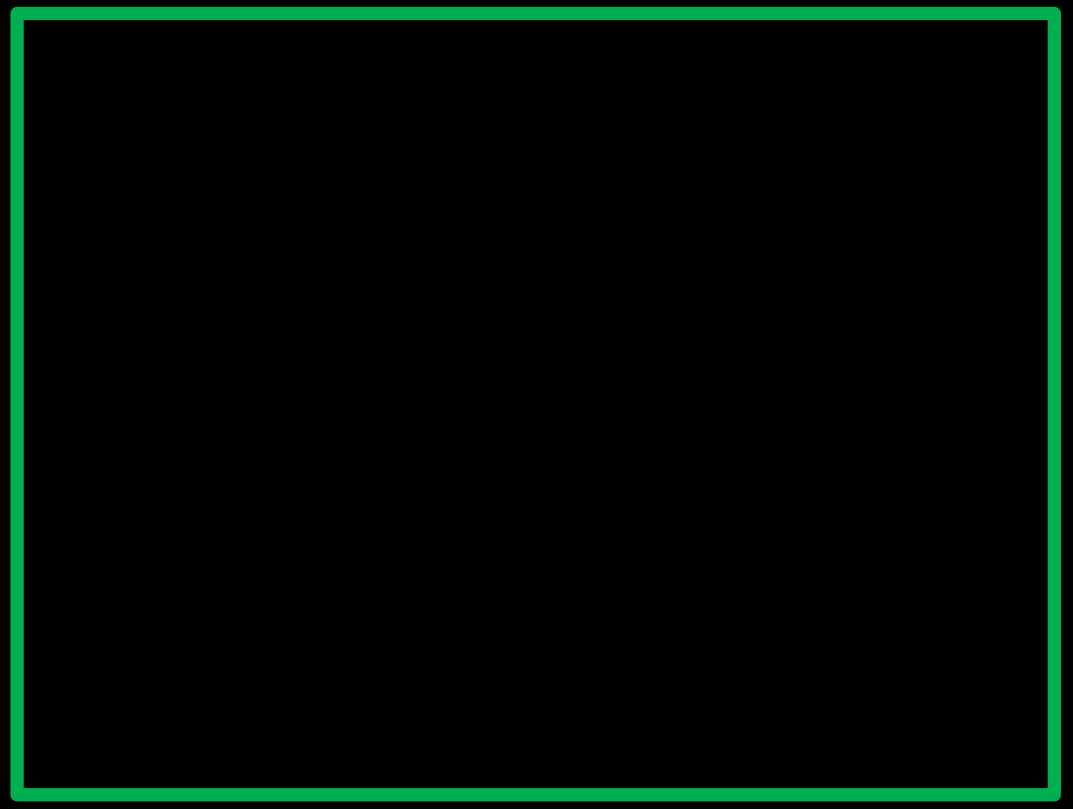


The PPU – Picture Processing Unit

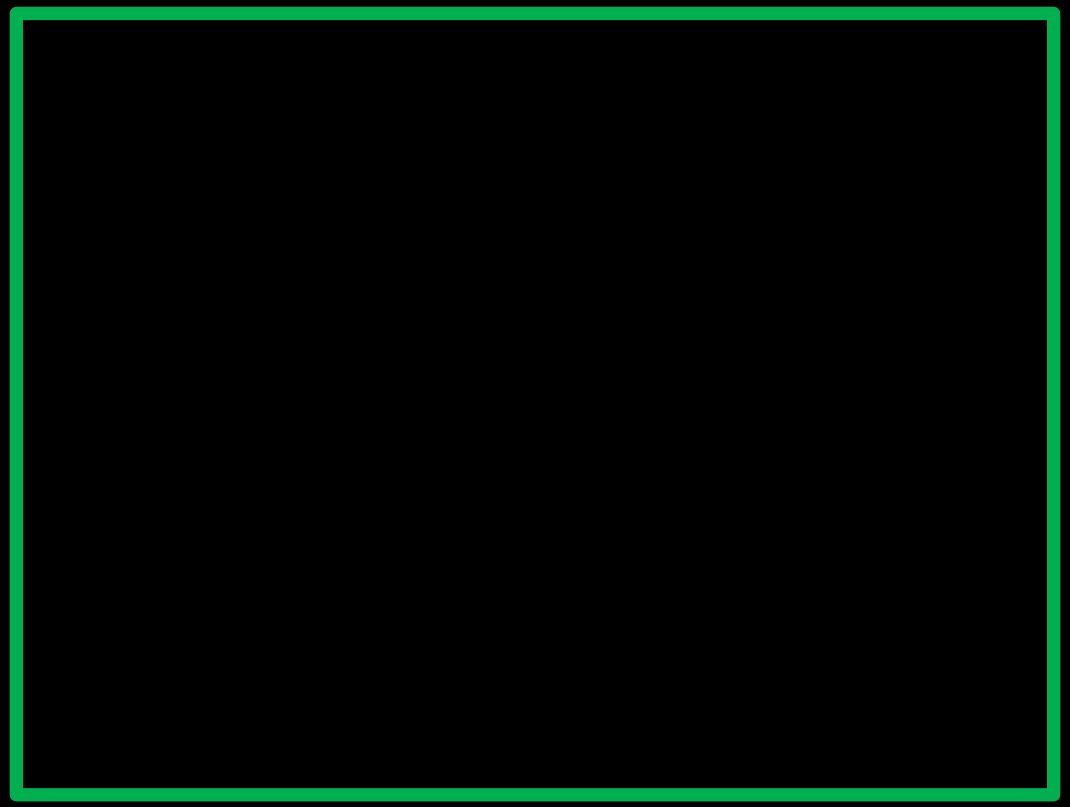


The APU – Audio Processing Unit





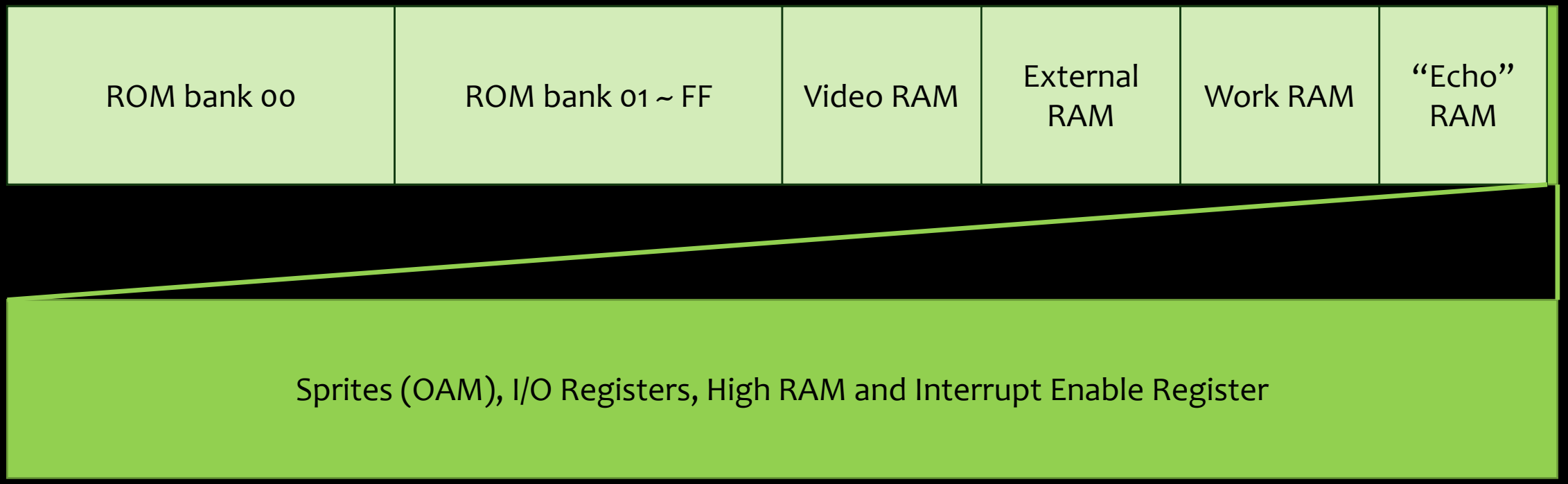
The APU – Audio Processing Unit



The APU – Audio Processing Unit

The APU – Audio Processing Unit

Memory layout



Booting a game

```
                LD SP,$fffe
                XOR A
                LD HL,$9fff
Addr_0007:
                LD (HL-),A
                BIT 7,H
                JR NZ, Addr_0007

                LD HL,$ff26
                LD C,$11
                LD A,$80
                LD (HL-),A
                LD ($FF00+C),A
                INC C
                LD A,$f3
                LD ($FF00+C),A
                LD (HL-),A
                LD A,$77
                LD (HL),A

                LD A,$fc
                LD ($FF00+$47),A

                LD DE,$0104
                LD HL,$8010
Addr_0027:
                LD A,(DE)
                CALL $0095
                CALL $0096
                INC DE
                LD A,E
                CP $34
                JR NZ, Addr_0027

                LD DE,$00d8
                LD B,$08
Addr_0039:
                LD A,(DE)
                INC DE
                LD (HL+),A
                INC HL
                DEC B
                JR NZ, Addr_0039

                LD A,$19
                LD ($9910),A
                LD HL,$992f
Addr_0048:
                LD C,$0c
Addr_004A:
                DEC A
                JR Z, Addr_0055
                LD (HL-),A
                DEC C
                JR NZ, Addr_004A
                LD L,$0f
                JR Addr_0048
Addr_0055:
                LD H,A
                LD A,$64
                LD D,A
                LD ($FF00+$42),A
                LD A,$91
                LD ($FF00+$40),A
                INC B
Addr_0060:
                LD E,$02
Addr_0062:
                LD C,$0c
Addr_0064:
                LD A,($FF00+$44)
                CP $90
                JR NZ, Addr_0064
                DEC C
                JR NZ, Addr_0064
                DEC E
                JR NZ, Addr_0062

                LD C,$13
                INC H
                LD A,H
                LD E,$83
                CP $62
                JR Z, Addr_0080
                LD E,$c1
                CP $64
                JR NZ, Addr_0086
Addr_0080:
                LD A,E
                LD ($FF00+C),A
                INC C
                LD A,$87
                LD ($FF00+C),A
Addr_0086:
                LD A,($FF00+$42)
                SUB B
                LD ($FF00+$42),A
                DEC D
                JR NZ, Addr_0060
                DEC B
                JR NZ, Addr_00E0
                LD D,$20
                JR Addr_0060
                LD C,A
                LD B,$04
Addr_0098:
                PUSH BC
                RL C
                RLA
                POP BC
                RL C
                RLA
                DEC B
                JR NZ, Addr_0098
                LD (HL+),A
                INC HL
                LD (HL+),A
                INC HL
                RET
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
                LD HL,$0104
                LD DE,$00a8
Addr_00E6:
                LD A,(DE)
                INC DE
                CP (HL)
                JR NZ,$fe
                INC HL
                LD A,L
                CP $34
                JR NZ, Addr_00E6
                LD B,$19
                LD A,B
Addr_00F4:
                ADD (HL)
                INC HL
                DEC B
                JR NZ, Addr_00F4
                ADD (HL)
                JR NZ,$fe
                LD A,$01
                LD ($FF00+$50),A
```

Booting a game

RAM initializer

```
                LD SP,$fffe
                XOR A
                LD HL,$9fff
Addr_0000:      LD (HL-),A
                BIT 7,H
                JR NZ, Addr_0007

                LD HL,$ff26
                LD C,$11
                LD A,$80
                LD (HL-),A
                LD ($FF00+C),A
                INC C
                LD A,$f3
                LD ($FF00+C),A
                LD (HL-),A
                LD A,$77
                LD (HL),A

                LD A,$fc
                LD ($FF00+$47),A

                LD DE,$0104
                LD HL,$8010
Addr_0027:
                LD A,(DE)
                CALL $0095
                CALL $0096
                INC DE
                LD A,E
                CP $34
                JR NZ, Addr_0027

                LD DE,$00d8
                LD B,$08
Addr_0039:
                LD A,(DE)
                INC DE
                LD (HL+),A
                INC HL
                DEC B
                JR NZ, Addr_0039

                LD A,$19
                LD ($9910),A
                LD HL,$992f
Addr_0048:
                LD C,$0c
Addr_004A:
                DEC A
                JR Z, Addr_0055
                LD (HL-),A
                DEC C
                JR NZ, Addr_004A
                LD L,$0f
                JR Addr_0048

Addr_0055:
                LD H,A
                LD A,$64
                LD D,A
                LD ($FF00+$42),A
                LD A,$91
                LD ($FF00+$40),A
                INC B
Addr_0060:
                LD E,$02
Addr_0062:
                LD C,$0c
Addr_0064:
                LD A,($FF00+$44)
                CP $90
                JR NZ, Addr_0064
                DEC C
                JR NZ, Addr_0064
                DEC E
                JR NZ, Addr_0062

                LD C,$13
                INC H
                LD A,H
                LD E,$83
                CP $62
                JR Z, Addr_0080
                LD E,$c1
                CP $64
                JR NZ, Addr_0086
Addr_0080:
                LD A,E
                LD ($FF00+C),A
                INC C
                LD A,$87
                LD ($FF00+C),A
Addr_0086:
                LD A,($FF00+$42)
                SUB B
                LD ($FF00+$42),A
                DEC D
                JR NZ, Addr_0060
                DEC B
                JR NZ, Addr_00E0
                LD D,$20
                JR Addr_0060

                LD C,A
                LD B,$04
Addr_0098:
                PUSH BC
                RL C
                RLA
                POP BC
                RL C
                RLA
                DEC B
                JR NZ, Addr_0098
                LD (HL+),A
                INC HL
                LD (HL+),A
                INC HL
                RET
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
                LD HL,$0104
                LD DE,$00a8
Addr_00E6:
                LD A,(DE)
                INC DE
                CP (HL)
                JR NZ,$fe
                INC HL
                LD A,L
                CP $34
                JR NZ, Addr_00E6
                LD B,$19
                LD A,B
Addr_00F4:
                ADD (HL)
                INC HL
                DEC B
                JR NZ, Addr_00F4
                ADD (HL)
                JR NZ,$fe
                LD A,$01
                LD ($FF00+$50),A
```

Booting a game

RAM initializer

Sound initializer

```
                LD SP,$fffe
                XOR A
                LD HL,$9fff
Addr_0000:
                LD (HL-),A
                BIT 7,H
                JR NZ, Addr_0007

                LD HL,$ff26
                LD C,$11
                LD A,$80
                LD (HL-),A
                LD ($FF00+C),A
                INC C
                LD A,$f3
                LD ($FF00+C),A
                LD (HL-),A
                LD A,$77
                LD (HL),A

                LD A,$fc
                LD ($FF00+$47),A

                LD DE,$0104
                LD HL,$8010
Addr_0027:
                LD A,(DE)
                CALL $0095
                CALL $0096
                INC DE
                LD A,E
                CP $34
                JR NZ, Addr_0027

                LD DE,$00d8
                LD B,$08
Addr_0039:
                LD A,(DE)
                INC DE
                LD (HL+),A
                INC HL
                DEC B
                JR NZ, Addr_0039

                LD A,$19
                LD ($9910),A
                LD HL,$992f
Addr_0048:
                LD C,$0c
Addr_004A:
                DEC A
                JR Z, Addr_0055
                LD (HL-),A
                DEC C
                JR NZ, Addr_004A
                LD L,$0f
                JR Addr_0048

Addr_0055:
                LD H,A
                LD A,$64
                LD D,A
                LD ($FF00+$42),A
                LD A,$91
                LD ($FF00+$40),A
                INC B
Addr_0060:
                LD E,$02
Addr_0062:
                LD C,$0c
Addr_0064:
                LD A,($FF00+$44)
                CP $90
                JR NZ, Addr_0064
                DEC C
                JR NZ, Addr_0064
                DEC E
                JR NZ, Addr_0062

                LD C,$13
                INC H
                LD A,H
                LD E,$83
                CP $62
                JR Z, Addr_0080
                LD E,$c1
                CP $64
                JR NZ, Addr_0086
Addr_0080:
                LD A,E
                LD ($FF00+C),A
                INC C
                LD A,$87
                LD ($FF00+C),A
Addr_0086:
                LD A,($FF00+$42)
                SUB B
                LD ($FF00+$42),A
                DEC D
                JR NZ, Addr_0060
                DEC B
                JR NZ, Addr_00E0
                LD D,$20
                JR Addr_0060

                LD C,A
                LD B,$04
Addr_0098:
                PUSH BC
                RL C
                RLA
                POP BC
                RL C
                RLA
                DEC B
                JR NZ, Addr_0098
                LD (HL+),A
                INC HL
                LD (HL+),A
                INC HL
                RET
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
                LD HL,$0104
                LD DE,$00a8
Addr_00E6:
                LD A,(DE)
                INC DE
                CP (HL)
                JR NZ,$fe
                INC HL
                LD A,L
                CP $34
                JR NZ, Addr_00E6
                LD B,$19
                LD A,B
Addr_00F4:
                ADD (HL)
                INC HL
                DEC B
                JR NZ, Addr_00F4
                ADD (HL)
                JR NZ,$fe
                LD A,$01
                LD ($FF00+$50),A
```

Booting a game

RAM initializer

Sound initializer

Set up Nintendo logo

```
Addr_0000:
                LD SP,$fffe
                XOR A
                LD HL,$9fff
Addr_0004:
                LD (HL-),A
                BIT 7,H
                JR NZ, Addr_0007

                LD HL,$ff26
                LD C,$11
                LD A,$80
                LD (HL-),A
                LD ($FF00+C),A
                INC C
                LD A,$f3
                LD ($FF00+C),A
                LD (HL-),A
                LD A,$77
                LD (HL),A

                LD A,$fc
                LD ($FF00+$47),A

                LD DE,$0104
                LD HL,$8010
Addr_0027:
                LD A,(DE)
                CALL $0095
                CALL $0096
                INC DE
                LD A,E
                CP $34
                JR NZ, Addr_0027

                LD DE,$00d8
                LD B,$08
Addr_0039:
                LD A,(DE)
                INC DE
                LD (HL+),A
                INC HL
                DEC B
                JR NZ, Addr_0039

                LD A,$19
                LD ($9910),A
                LD HL,$992f
Addr_0048:
                LD C,$0c
Addr_004A:
                DEC A
                JR Z, Addr_0055
                LD (HL-),A
                DEC C
                JR NZ, Addr_004A
                LD L,$0f
                JR Addr_0048

Addr_0055:
                LD H,A
                LD A,$64
                LD D,A
                LD ($FF00+$42),A
                LD A,$91
                LD ($FF00+$40),A
                INC B
Addr_0060:
                LD E,$02
Addr_0062:
                LD C,$0c
Addr_0064:
                LD A,($FF00+$44)
                CP $90
                JR NZ, Addr_0064
                DEC C
                JR NZ, Addr_0064
                DEC E
                JR NZ, Addr_0062

                LD C,$13
                INC H
                LD A,H
                LD E,$83
                CP $62
                JR Z, Addr_0080
                LD E,$c1
                CP $64
                JR NZ, Addr_0086
Addr_0080:
                LD A,E
                LD ($FF00+C),A
                INC C
                LD A,$87
                LD ($FF00+C),A
Addr_0086:
                LD A,($FF00+$42)
                SUB B
                LD ($FF00+$42),A
                DEC D
                JR NZ, Addr_0060
                DEC B
                JR NZ, Addr_00E0
                LD D,$20
                JR Addr_0060

                LD C,A
                LD B,$04
Addr_0098:
                PUSH BC
                RL C
                RLA
                POP BC
                RL C
                RLA
                DEC B
                JR NZ, Addr_0098
                LD (HL+),A
                INC HL
                LD (HL+),A
                INC HL
                RET
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
                LD HL,$0104
                LD DE,$00a8
Addr_00E6:
                LD A,(DE)
                INC DE
                CP (HL)
                JR NZ,$fe
                INC HL
                LD A,L
                CP $34
                JR NZ, Addr_00E6
                LD B,$19
                LD A,B
Addr_00F4:
                ADD (HL)
                INC HL
                DEC B
                JR NZ, Addr_00F4
                ADD (HL)
                JR NZ,$fe
                LD A,$01
                LD ($FF00+$50),A
```

Booting a game

```
LD SP,$fffe
XOR A
LD HL,$9fff
Addr_0007:
LD (HL-),A
BIT 7,H
JR NZ, Addr_0007
```

RAM initializer

```
LD HL,$ff26
LD C,$11
LD A,$80
LD (HL-),A
LD ($FF00+C),A
INC C
LD A,$f3
LD ($FF00+C),A
LD (HL-),A
LD A,$77
LD (HL),A
```

Sound initializer

```
LD A,$fc
LD ($FF00+$47),A

LD DE,$0104
LD HL,$8010
Addr_0027:
LD A,(DE)
CALL $0095
CALL $0096
INC DE
LD A,E
CP $34
JR NZ, Addr_0027

LD DE,$00d8
LD B,$08
Addr_0039:
LD A,(DE)
INC DE
LD (HL+),A
INC HL
DEC B
JR NZ, Addr_0039

LD A,$19
LD ($9910),A
LD HL,$992f
Addr_0048:
LD C,$0c
Addr_004A:
DEC A
JR Z, Addr_0055
LD (HL-),A
DEC C
JR NZ, Addr_004A
LD L,$0f
JR Addr_0048
```

Set up Nintendo logo

```
Addr_0055:
LD H,A
LD A,$64
LD D,A
LD ($FF00+$42),A
LD A,$91
LD ($FF00+$40),A
INC B
Addr_0060:
LD E,$02
Addr_0062:
LD C,$0c
Addr_0064:
LD A,($FF00+$44)
CP $90
JR NZ, Addr_0064
DEC C
JR NZ, Addr_0064
DEC E
JR NZ, Addr_0062

LD C,$13
INC H
LD A,H
LD E,$83
CP $62
JR Z, Addr_0080
LD E,$c1
CP $64
JR NZ, Addr_0086
Addr_0080:
LD A,E
LD ($FF00+C),A
INC C
LD A,$87
LD ($FF00+C),A
Addr_0086:
LD A,($FF00+$42)
SUB B
LD ($FF00+$42),A
DEC D
JR NZ, Addr_0060
DEC B
JR NZ, Addr_00E0
LD D,$20
JR Addr_0060
```

```
LD C,A
LD B,$04
Addr_0098:
PUSH BC
RL C
RLA
POP BC
RL C
RLA
DEC B
JR NZ, Addr_0098
LD (HL+),A
INC HL
LD (HL+),A
INC HL
RET
```

Decode Nintendo logo

```
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
LD HL,$0104
LD DE,$00a8
Addr_00E6:
LD A,(DE)
INC DE
CP (HL)
JR NZ,$fe
INC HL
LD A,L
CP $34
JR NZ, Addr_00E6
LD B,$19
LD A,B
Addr_00F4:
ADD (HL)
INC HL
DEC B
JR NZ, Addr_00F4
ADD (HL)
JR NZ,$fe
LD A,$01
LD ($FF00+$50),A
```

Booting a game

```
                LD SP,$fffe
                XOR A
                LD HL,$9fff
Addr_0007:
                LD (HL-),A
                BIT 7,H
                JR NZ, Addr_0007

                LD HL,$ff26
                LD C,$11
                LD A,$80
                LD (HL-),A
                LD ($FF00+C),A
                INC C
                LD A,$f3
                LD ($FF00+C),A
                LD (HL-),A
                LD A,$77
                LD (HL),A

                LD A,$fc
                LD ($FF00+$47),A

                LD DE,$0104
                LD HL,$8010
Addr_0027:
                LD A,(DE)
                CALL $0095
                CALL $0096
                INC DE
                LD A,E
                CP $34
                JR NZ, Addr_0027

                LD DE,$00d8
                LD B,$08
Addr_0039:
                LD A,(DE)
                INC DE
                LD (HL+),A
                INC HL
                DEC B
                JR NZ, Addr_0039

                LD A,$19
                LD ($9910),A
                LD HL,$992f
Addr_0048:
                LD C,$0c
Addr_004A:
                DEC A
                JR Z, Addr_0055
                LD (HL-),A
                DEC C
                JR NZ, Addr_004A
                LD L,$0f
                JR Addr_0048
```

RAM initializer

Sound initializer

Set up Nintendo logo

```
Addr_0055:
                LD H,A
                LD A,$64
                LD D,A
                LD ($FF00+$42),A
                LD A,$91
                LD ($FF00+$40),A
                INC B
Addr_0060:
                LD E,$02
Addr_0062:
                LD C,$0c
Addr_0064:
                LD A,($FF00+$44)
                JR NZ, Addr_0064
                DEC C
                JR NZ, Addr_0064
                DEC E
                JR NZ, Addr_0062

                LD C,$13
                INC H
                LD A,H
                LD E,$83
                CP $62
                JR Z, Addr_0080
                LD E,$c1
                CP $64
                JR NZ, Addr_0086
Addr_0080:
                LD A,E
                LD ($FF00+C),A
                INC C
                LD A,$87
                LD ($FF00+C),A
Addr_0086:
                LD A,($FF00+$42)
                SUB B
                LD ($FF00+$42),A
                DEC D
                JR NZ, Addr_0060
                DEC E
                JR NZ, Addr_00E0
                LD D,$20
                JR Addr_0060
```

Scroll logo

Scroll logo

```
                LD C,A
                LD B,$04
Addr_0098:
                PUSH BC
                RL C
                RLA
                POP BC
                RL C
                RLA
                DEC B
                JR NZ, Addr_0098
                LD (HL+),A
                INC HL
                LD (HL+),A
                INC HL
                RET

Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
                LD HL,$0104
                LD DE,$00a8
Addr_00E6:
                LD A,(DE)
                INC DE
                CP (HL)
                JR NZ,$fe
                INC HL
                LD A,L
                CP $34
                JR NZ, Addr_00E6
                LD B,$19
                LD A,B
Addr_00F4:
                ADD (HL)
                INC HL
                DEC B
                JR NZ, Addr_00F4
                ADD (HL)
                JR NZ,$fe
                LD A,$01
                LD ($FF00+$50),A
```

Decode Nintendo logo

Booting a game

RAM initializer

```
LD SP,$fffe
XOR A
LD HL,$9fff
LD (HL-),A
BIT 7,H
JR NZ, Addr_0007
```

Sound initializer

```
LD HL,$ff26
LD C,$11
LD A,$80
LD (HL-),A
LD ($FF00+C),A
INC C
LD A,$f3
LD ($FF00+C),A
LD (HL-),A
LD A,$77
LD (HL),A
```

Set up Nintendo logo

```
LD A,$fc
LD ($FF00+$47),A

LD DE,$0104
LD HL,$8010
Addr_0027:
LD A,(DE)
CALL $0095
CALL $0096
INC DE
LD A,E
CP $34
JR NZ, Addr_0027

LD DE,$00d8
LD B,$08
Addr_0039:
LD A,(DE)
INC DE
LD (HL+),A
INC HL
DEC B
JR NZ, Addr_0039

LD A,$19
LD ($9910),A
LD HL,$992f
Addr_0048:
LD C,$0c
Addr_004A:
DEC A
JR Z, Addr_0055
LD (HL-),A
DEC C
JR NZ, Addr_004A
LD L,$0f
JR Addr_0048
```

Scroll logo

```
Addr_0055:
LD H,A
LD A,$64
LD D,A
LD ($FF00+$42),A
LD A,$91
LD ($FF00+$40),A
INC B
Addr_0060:
LD E,$02
Addr_0062:
LD C,$0c
Addr_0064:
LD A,($FF00+$44)
JR NZ, Addr_0064
DEC C
JR NZ, Addr_0064
DEC E
JR NZ, Addr_0062

LD C,$13
INC H
LD A,H
LD E,$83
CP $62
JR Z, Addr_0080
LD E,$c1
CP $64
JR NZ, Addr_0086
```

Play sound

```
Addr_0080:
LD A,E
LD ($FF00+C),A
INC C
LD A,$87
LD ($FF00+C),A
```

Scroll logo

```
Addr_0086:
LD A,($FF00+$42)
SUB B
LD ($FF00+$42),A
DEC D
JR NZ, Addr_0060
DEC E
JR NZ, Addr_00E0
LD D,$20
JR Addr_0060
```

Decode Nintendo logo

```
LD C,A
LD B,$04
Addr_0098:
PUSH BC
RL C
RLA
POP BC
RL C
RLA
DEC B
JR NZ, Addr_0098
LD (HL+),A
INC HL
LD (HL+),A
INC HL
RET
```

```
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E

Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C

Addr_00E0:
LD HL,$0104
LD DE,$00a8
Addr_00E6:
LD A,(DE)
INC DE
CP (HL)
JR NZ,$fe
INC HL
LD A,L
CP $34
JR NZ, Addr_00E6
LD B,$19
LD A,B
Addr_00F4:
ADD (HL)
INC HL
DEC B
JR NZ, Addr_00F4
ADD (HL)
JR NZ,$fe
LD A,$01
LD ($FF00+$50),A
```

Booting a game

RAM initializer

```
LD SP,$fffe
XOR A
LD HL,$9fff
LD (HL-),A
BIT 7,H
JR NZ, Addr_0007
```

Sound initializer

```
LD HL,$ff26
LD C,$11
LD A,$80
LD (HL-),A
LD ($FF00+C),A
INC C
LD A,$f3
LD ($FF00+C),A
LD (HL-),A
LD A,$77
LD (HL),A
```

Set up Nintendo logo

```
LD A,$fc
LD ($FF00+$47),A

LD DE,$0104
LD HL,$8010
Addr_0027:

LD A,(DE)
CALL $0095
CALL $0096
INC DE
LD A,E
CP $34
JR NZ, Addr_0027

LD DE,$00d8
LD B,$08
Addr_0039:
LD A,(DE)
INC DE
LD (HL+),A
INC HL
DEC B
JR NZ, Addr_0039

LD A,$19
LD ($9910),A
LD HL,$992f
Addr_0048:
LD C,$0c
Addr_004A:
DEC A
JR Z, Addr_0055
LD (HL-),A
DEC C
JR NZ, Addr_004A
LD L,$0f
JR Addr_0048
```

Scroll logo

```
Addr_0055:
LD H,A
LD A,$64
LD D,A
LD ($FF00+$42),A
LD A,$91
LD ($FF00+$40),A
INC B
Addr_0060:
LD E,$02
Addr_0062:
LD C,$0c
Addr_0064:
LD A,($FF00+$44)
JR NZ, Addr_0064
DEC C
JR NZ, Addr_0064
DEC E
JR NZ, Addr_0062

LD C,$13
INC H
LD A,H
LD E,$83
CP $62
JR Z, Addr_0080
LD E,$c1
CP $64
JR NZ, Addr_0086
```

Play sound

```
Addr_0080:
LD A,E
LD ($FF00+C),A
INC C
LD A,$87
LD ($FF00+C),A
```

Scroll logo

```
Addr_0086:
LD A,($FF00+$42)
SUB B
LD ($FF00+$42),A
DEC D
JR NZ, Addr_0060
DEC E
JR NZ, Addr_00E0
LD D,$20
JR Addr_0060
```

Decode Nintendo logo

```
LD C,A
LD B,$04
Addr_0098:
PUSH BC
RL C
RLA
POP BC
RL C
DEC B
JR NZ, Addr_0098
LD (HL+),A
INC HL
LD (HL+),A
INC HL
RET
```

Nintendo logo data

```
Addr_00A8:
;Nintendo Logo
.DB $CE,$ED,$66,$66,$CC,$0D,$00,$0B,$03,$73,$00,$83,$00,$0C,$00,$0D
.DB $00,$08,$11,$1F,$88,$89,$00,$0E,$DC,$CC,$6E,$E6,$DD,$DD,$D9,$99
.DB $BB,$BB,$67,$63,$6E,$0E,$EC,$CC,$DD,$DC,$99,$9F,$BB,$B9,$33,$3E
Addr_00D8:
;More video data
.DB $3C,$42,$B9,$A5,$B9,$A5,$42,$3C
```

Compare logo

```
Addr_00E0:
LD HL,$0104
LD DE,$00a8
Addr_00E6:
LD A,(DE)
INC DE
JR NZ,$fe
INC HL
LD A,L
CP $34
JR NZ, Addr_00E6
LD B,$19
LD A,B
Addr_00F4:
ADD (HL)
INC HL
DEC B
JR NZ, Addr_00F4
ADD (HL)
JR NZ,$fe
LD A,$01
LD ($FF00+$50),A
```

Booting a game

RAM initializer

Sound initializer

Set up Nintendo logo

Scroll logo

Play sound

Scroll logo

Decode Nintendo logo

Nintendo logo data

Compare logo

Checksum calculation

Disable boot ROM

Emulating the Game Boy

Emulating the Game Boy

Game Boy CPU (SM83) instruction set (JSON)

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NOP 1 4 - - - -	LD BC, d16 3 12 - - - -	LD (BC), A 1 8 - - - -	INC BC 1 8 - - - -	INC B 1 4 Z 0 H -	DEC B 1 4 Z 1 H -	LD B, d8 2 8 - - - -	RLCA 1 4 0 0 0 C	LD (a16), SP 3 20 - - - -	ADD HL, BC 1 8 - 0 H C	LD A, (BC) 1 8 - - - -	DEC BC 1 8 - - - -	INC C 1 4 Z 0 H -	DEC C 1 4 Z 1 H -	LD C, d8 2 8 - - - -	RRCA 1 4 0 0 0 C
1x	STOP d8 2 4 - - - -	LD DE, d16 3 12 - - - -	LD (DE), A 1 8 - - - -	INC DE 1 8 - - - -	INC D 1 4 Z 0 H -	DEC D 1 4 Z 1 H -	LD D, d8 2 8 - - - -	RLA 1 4 0 0 0 C	JR r8 2 12 - - - -	ADD HL, DE 1 8 - 0 H C	LD A, (DE) 1 8 - - - -	DEC DE 1 8 - - - -	INC E 1 4 Z 0 H -	DEC E 1 4 Z 1 H -	LD E, d8 2 8 - - - -	RRA 1 4 0 0 0 C
2x	JR NZ, r8 2 12/8 - - - -	LD HL, d16 3 12 - - - -	LD (HL+), A 1 8 - - - -	INC HL 1 8 - - - -	INC H 1 4 Z 0 H -	DEC H 1 4 Z 1 H -	LD H, d8 2 8 - - - -	DAA 1 4 Z - 0 C	JR Z, r8 2 12/8 - - - -	ADD HL, HL 1 8 - 0 H C	LD A, (HL+) 1 8 - - - -	DEC HL 1 8 - - - -	INC L 1 4 Z 0 H -	DEC L 1 4 Z 1 H -	LD L, d8 2 8 - - - -	CPL 1 4 - 1 1 -
3x	JR NC, r8 2 12/8 - - - -	LD SP, d16 3 12 - - - -	LD (HL-), A 1 8 - - - -	INC SP 1 8 - - - -	INC (HL) 1 12 Z 0 H -	DEC (HL) 1 12 Z 1 H -	LD (HL), d8 2 12 - - - -	SCF 1 4 - 0 0 1	JR C, r8 2 12/8 - - - -	ADD HL, SP 1 8 - 0 H C	LD A, (HL-) 1 8 - - - -	DEC SP 1 8 - - - -	INC A 1 4 Z 0 H -	DEC A 1 4 Z 1 H -	LD A, d8 2 8 - - - -	CCF 1 4 - 0 0 C
4x	LD B, B 1 4 - - - -	LD B, C 1 4 - - - -	LD B, D 1 4 - - - -	LD B, E 1 4 - - - -	LD B, H 1 4 - - - -	LD B, L 1 4 - - - -	LD B, (HL) 1 8 - - - -	LD B, A 1 4 - - - -	LD C, B 1 4 - - - -	LD C, C 1 4 - - - -	LD C, D 1 4 - - - -	LD C, E 1 4 - - - -	LD C, H 1 4 - - - -	LD C, L 1 4 - - - -	LD C, (HL) 1 8 - - - -	LD C, A 1 4 - - - -
5x	LD D, B 1 4 - - - -	LD D, C 1 4 - - - -	LD D, D 1 4 - - - -	LD D, E 1 4 - - - -	LD D, H 1 4 - - - -	LD D, L 1 4 - - - -	LD D, (HL) 1 8 - - - -	LD D, A 1 4 - - - -	LD E, B 1 4 - - - -	LD E, C 1 4 - - - -	LD E, D 1 4 - - - -	LD E, E 1 4 - - - -	LD E, H 1 4 - - - -	LD E, L 1 4 - - - -	LD E, (HL) 1 8 - - - -	LD E, A 1 4 - - - -
6x	LD H, B 1 4 - - - -	LD H, C 1 4 - - - -	LD H, D 1 4 - - - -	LD H, E 1 4 - - - -	LD H, H 1 4 - - - -	LD H, L 1 4 - - - -	LD H, (HL) 1 8 - - - -	LD H, A 1 4 - - - -	LD L, B 1 4 - - - -	LD L, C 1 4 - - - -	LD L, D 1 4 - - - -	LD L, E 1 4 - - - -	LD L, H 1 4 - - - -	LD L, L 1 4 - - - -	LD L, (HL) 1 8 - - - -	LD L, A 1 4 - - - -
7x	LD (HL), B 1 8 - - - -	LD (HL), C 1 8 - - - -	LD (HL), D 1 8 - - - -	LD (HL), E 1 8 - - - -	LD (HL), H 1 8 - - - -	LD (HL), L 1 8 - - - -	HALT 1 4 - - - -	LD (HL), A 1 8 - - - -	LD A, B 1 4 - - - -	LD A, C 1 4 - - - -	LD A, D 1 4 - - - -	LD A, E 1 4 - - - -	LD A, H 1 4 - - - -	LD A, L 1 4 - - - -	LD A, (HL) 1 8 - - - -	LD A, A 1 4 - - - -
8x	ADD A, B 1 4 Z 0 H C	ADD A, C 1 4 Z 0 H C	ADD A, D 1 4 Z 0 H C	ADD A, E 1 4 Z 0 H C	ADD A, H 1 4 Z 0 H C	ADD A, L 1 4 Z 0 H C	ADD A, (HL) 1 8 Z 0 H C	ADD A, A 1 4 Z 0 H C	ADC A, B 1 4 Z 0 H C	ADC A, C 1 4 Z 0 H C	ADC A, D 1 4 Z 0 H C	ADC A, E 1 4 Z 0 H C	ADC A, H 1 4 Z 0 H C	ADC A, L 1 4 Z 0 H C	ADC A, (HL) 1 8 Z 0 H C	ADC A, A 1 4 Z 0 H C
9x	SUB B 1 4 Z 1 H C	SUB C 1 4 Z 1 H C	SUB D 1 4 Z 1 H C	SUB E 1 4 Z 1 H C	SUB H 1 4 Z 1 H C	SUB L 1 4 Z 1 H C	SUB (HL) 1 8 Z 1 H C	SUB A 1 4 1 1 0 0	SBC A, B 1 4 Z 1 H C	SBC A, C 1 4 Z 1 H C	SBC A, D 1 4 Z 1 H C	SBC A, E 1 4 Z 1 H C	SBC A, H 1 4 Z 1 H C	SBC A, L 1 4 Z 1 H C	SBC A, (HL) 1 8 Z 1 H C	SBC A, A 1 4 Z 1 H C
Ax	AND B 1 4 Z 0 1 0	AND C 1 4 Z 0 1 0	AND D 1 4 Z 0 1 0	AND E 1 4 Z 0 1 0	AND H 1 4 Z 0 1 0	AND L 1 4 Z 0 1 0	AND (HL) 1 8 Z 0 1 0	AND A 1 4 Z 0 1 0	XOR B 1 4 Z 0 0 0	XOR C 1 4 Z 0 0 0	XOR D 1 4 Z 0 0 0	XOR E 1 4 Z 0 0 0	XOR H 1 4 Z 0 0 0	XOR L 1 4 Z 0 0 0	XOR (HL) 1 8 Z 0 0 0	XOR A 1 4 1 0 0 0
Bx	OR B 1 4 Z 0 0 0	OR C 1 4 Z 0 0 0	OR D 1 4 Z 0 0 0	OR E 1 4 Z 0 0 0	OR H 1 4 Z 0 0 0	OR L 1 4 Z 0 0 0	OR (HL) 1 8 Z 0 0 0	OR A 1 4 Z 0 0 0	CP B 1 4 Z 1 H C	CP C 1 4 Z 1 H C	CP D 1 4 Z 1 H C	CP E 1 4 Z 1 H C	CP H 1 4 Z 1 H C	CP L 1 4 Z 1 H C	CP (HL) 1 8 Z 1 H C	CP A 1 4 1 1 0 0
Cx	RET NZ 1 20/8 - - - -	POP BC 1 12 - - - -	JP NZ, a16 3 16/12 - - - -	JP a16 3 16 - - - -	CALL NZ, a16 3 24/12 - - - -	PUSH BC 1 16 - - - -	ADD A, d8 2 8 Z 0 H C	RST 00H 1 16 - - - -	RET Z 1 20/8 - - - -	RET 1 16 - - - -	JP Z, a16 3 16/12 - - - -	PREFIX 1 4 - - - -	CALL Z, a16 3 24/12 - - - -	CALL a16 3 24 - - - -	ADC A, d8 2 8 Z 0 H C	RST 08H 1 16 - - - -
Dx	RET NC 1 20/8 - - - -	POP DE 1 12 - - - -	JP NC, a16 3 16/12 - - - -	—	CALL NC, a16 3 24/12 - - - -	PUSH DE 1 16 - - - -	SUB d8 2 8 Z 1 H C	RST 10H 1 16 - - - -	RET C 1 20/8 - - - -	RETI 1 16 - - - -	JP C, a16 3 16/12 - - - -	—	CALL C, a16 3 24/12 - - - -	—	SBC A, d8 2 8 Z 1 H C	RST 18H 1 16 - - - -
Ex	LDH (a8), A 2 12 - - - -	POP HL 1 12 - - - -	LD (C), A 1 8 - - - -	—	—	PUSH HL 1 16 - - - -	AND d8 2 8 Z 0 1 0	RST 20H 1 16 - - - -	ADD SP, r8 2 16 0 0 H C	JP HL 1 4 - - - -	LD (a16), A 3 16 - - - -	—	—	—	XOR d8 2 8 Z 0 0 0	RST 28H 1 16 - - - -
Fx	LDH A, (a8) 2 12 - - - -	POP AF 1 12 Z N H C	LD A, (C) 1 8 - - - -	DI 1 4 - - - -	—	PUSH AF 1 16 - - - -	OR d8 2 8 Z 0 0 0	RST 30H 1 16 - - - -	LD HL, SP + r8 2 12 0 0 H C	LD SP, HL 1 8 - - - -	LD A, (a16) 3 16 - - - -	EI 1 4 - - - -	—	—	CP d8 2 8 Z 1 H C	RST 38H 1 16 - - - -

Emulating the Game Boy

ction set (JSON)

x7	x8	x9	xA	xB	
RLCA 1 4 0 0 0 C	LD (a16), SP 3 20 - - - -	ADD HL, BC 1 8 - 0 H C	LD A, (BC) 1 8 - - - -	DEC BC 1 8 - - - -	
RLA 1 4 0 0 0 C	JR r8 2 12 - - - -	ADD HL, DE 1 8 - 0 H C	LD A, (DE) 1 8 - - - -	DEC DE 1 8 - - - -	
DAA 1 4 Z - 0 C	JR Z, r8 2 12/8 - - - -	ADD HL, HL 1 8 - 0 H C	LD A, (HL+) 1 8 - - - -	DEC HL 1 8 - - - -	
SCF 1 4 - 0 0 1	JR C, r8 2 12/8 - - - -	ADD HL, SP 1 8 - 0 H C	LD A, (HL-) 1 8 - - - -	DEC SP 1 8 - - - -	
LD B, A 1 4 - - - -	LD C, B 1 4 - - - -	LD C, C 1 4 - - - -	LD C, D 1 4 - - - -	LD C, E 1 4 - - - -	

Emulating the Game Boy

```
[
   // ...
   "0x10": {
      "mnemonic": "STOP", // Hammertime!
      "bytes": 2,
      "cycles": [
         4
      ],
      "operands": [
         {
            "name": "d8",
            "bytes": 1,
            "immediate": true
         }
      ],
      "immediate": true,
      "flags": {
         "Z": "-",
         "N": "-",
         "H": "-",
         "C": "-"
      }
   },
   // ...
]
```

```
public void Run(CancellationToken cancellationToken)
{
    Registers.PC = 0x0000;
    _cycles = 0;
    var reduceTicksBy = 0;

    var lastTimer = DateTimeOffset.UtcNow.Ticks;
    var now = lastTimer;

    for(;;)
    {
        var opCode = OpCode.Get((Prefix << 8) + Memory.Read(Registers.PC));
        opCode.Execute(this);

        _cycles += opCode.Cycles(); // Number of cycles can depend on the execution of the action

        // check for interrupts, draw, i/o, ...
        if (_cycles ≥ DividerRegisterTicks)
        {
            // todo: if stop mode, do nothing
            Memory.IncreaseDividerRegister();
            reduceTicksBy = DividerRegisterTicks;
        }

        if (reduceTicksBy > 0)
        {
            _cycles -= reduceTicksBy;
            reduceTicksBy = 0;
        }

        if (cancellationToken.IsCancellationRequested)
        {
            break;
        }
    }
}
```

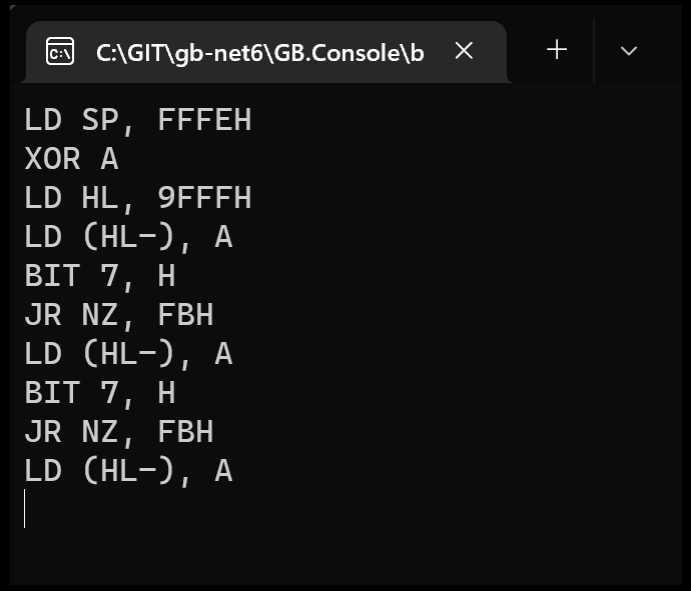
```
    public void Execute(Cpu cpu)
    {
        if (_opCode == 0xCB)
        {
            cpu.Prefix = 0xCB;
            cpu.Registers.PC += _bytes;
            return;
        }

#if DEBUG
        Console.WriteLine(ToString());
#endif

        var bytes = _bytes;
        if (cpu.Prefix == 0xCB)
        {
            bytes--;
            _reduceCyclesByFour = true;
        }
        cpu.Prefix = 0x00;

        switch (_mnemonic)
        {
            case "NOP": // No operation
                break;
            case "STOP": // Stop CPU
                // cpu.Stopped = true;
                break;
            case "HALT": // Halt CPU
                // cpu.Halted = true;
                break;
            case "DI": // Disable interrupts
                // cpu.InterruptsEnabled = false;
                break;
            case "EI": // Enable interrupts
                // cpu.InterruptsEnabled = true;
                break;
```

```
case "LD":
case "LDH":
    Load(cpu);
    break;
case "ADC": // Add with carry two registers, memory to register, direct to memory, ...
    Add(cpu, true);
    break;
case "ADD": // Add two registers, memory to register, direct to memory, ...
    Add(cpu, false);
    break;
case "AND": // AND two registers, memory to register, direct to memory, ...
    And(cpu);
    break;
case "CP": // Compare two registers, memory to register, direct to memory, ...
    Compare(cpu);
    break;
case "OR": // OR two registers, memory to register, direct to memory, ...
    Or(cpu);
    break;
case "SBC": // Subtract with carry two registers, memory to register, direct to memory, ...
    Subtract(cpu, true);
    break;
case "SUB": // Subtract two registers, memory to register, direct to memory, ...
    Subtract(cpu, false);
    break;
case "XOR": // XOR two registers, memory to register, direct to memory, ...
    Xor(cpu);
    break;
// Stack
case "PUSH":
    Push(cpu);
    break;
```

```
/// <summary>
/// LD target, source
/// LDH target, source
/// LDHL target, source
/// Where source or target can be:
/// * an 8 or 16-bit register
/// * an indirect address (register points to memory location)
/// * a direct address
/// * indexed address (I/O range FF00 + direct offset / register offset)
/// </summary>
/// <param name="cpu"></param>
[MethodImpl(MethodImplOptions.AggressiveInlining)]
1 reference | Wesley Cabus, 205 days ago | 1 author, 1 change
private void Load(Cpu cpu)
{
    var source = _operands[1];
    var sourceData = ReadSource(source, cpu);

    if (_operands.Length == 3) // LDHL SP,n
    {
        var sourceData2 = (sbyte)ReadSource(_operands[2], cpu);
        sourceData += sourceData2;
    }

    var target = _operands[0];
    WriteTarget(target, sourceData, cpu);

    if (_opCode == 0x00F8)
    {
        UpdateFlags(cpu, false, false, (sourceData & 0x10) != 0, (sourceData & 0x100) != 0);
    }
}
```

Did it work?

Did it work?


C:\GIT\gb-net6\GB.Console\b
LD SP, FFFEH
XOR A
LD HL, 9FFFH
LD (HL-), A
BIT 7, H
JR NZ, FBH
LD (HL-), A
BIT 7, H
JR NZ, FBH
LD (HL-), A

C:\GIT\gb-net6\GB.Console\b

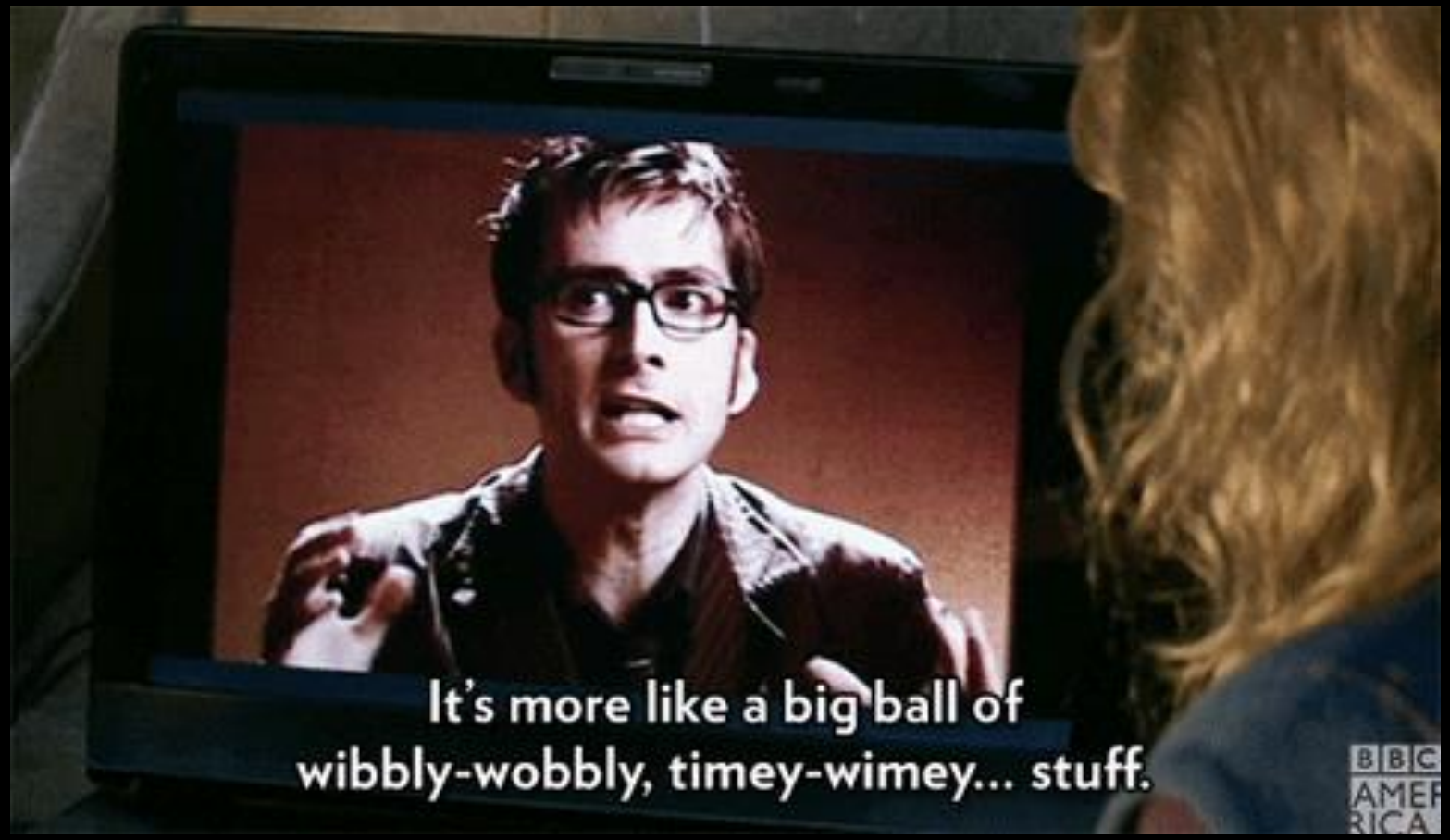
```
LD SP, FFFEH
XOR A
LD HL, 9FFFH
LD (HL-), A
BIT 7, H
JR NZ, FBH
LD (HL-), A
BIT 7, H
JR NZ, FBH
LD (HL-), A
```

```
        LD SP,$fffe
        XOR A
        LD HL,$9fff
Addr_0007:
        LD (HL-),A
        BIT 7,H
        JR NZ, Addr_0007
```

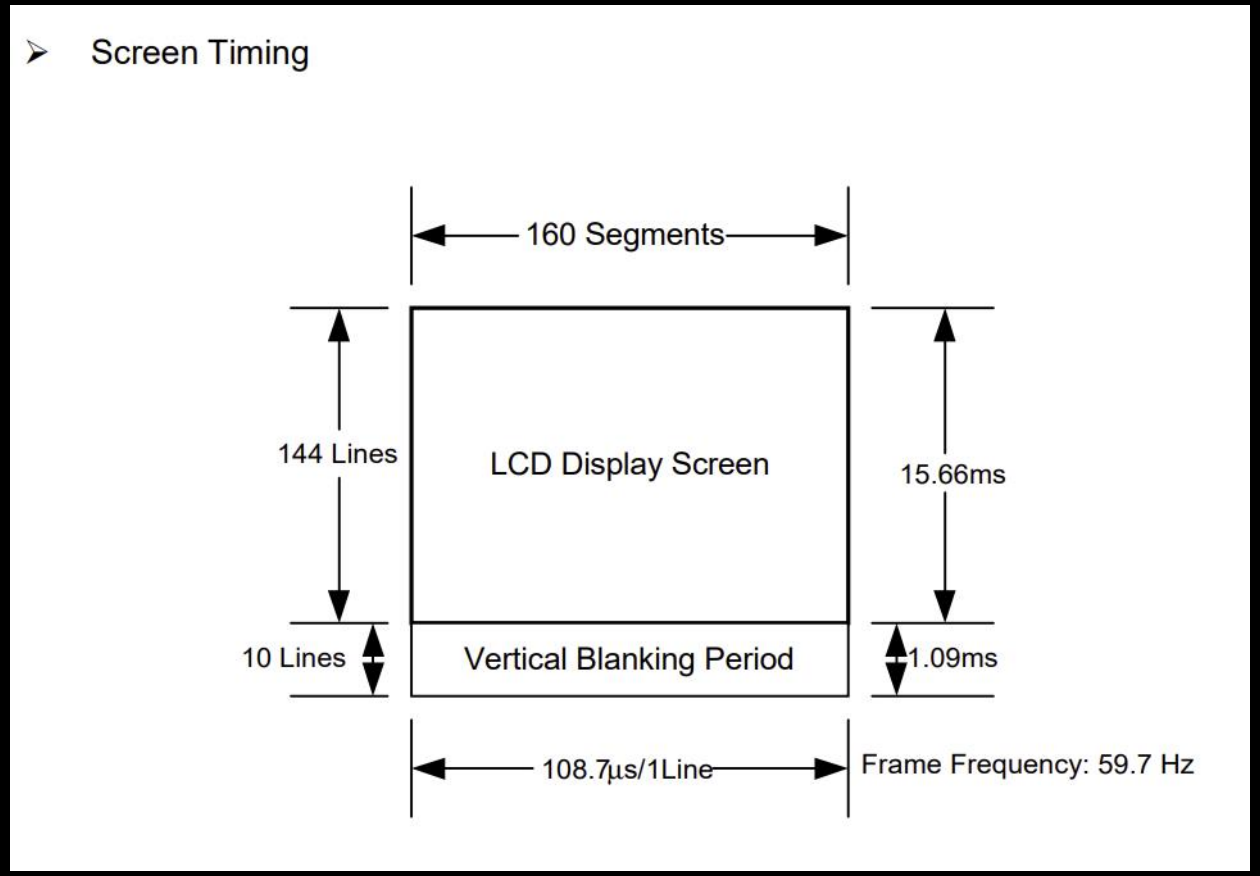
Next: graphics

Next: graphics

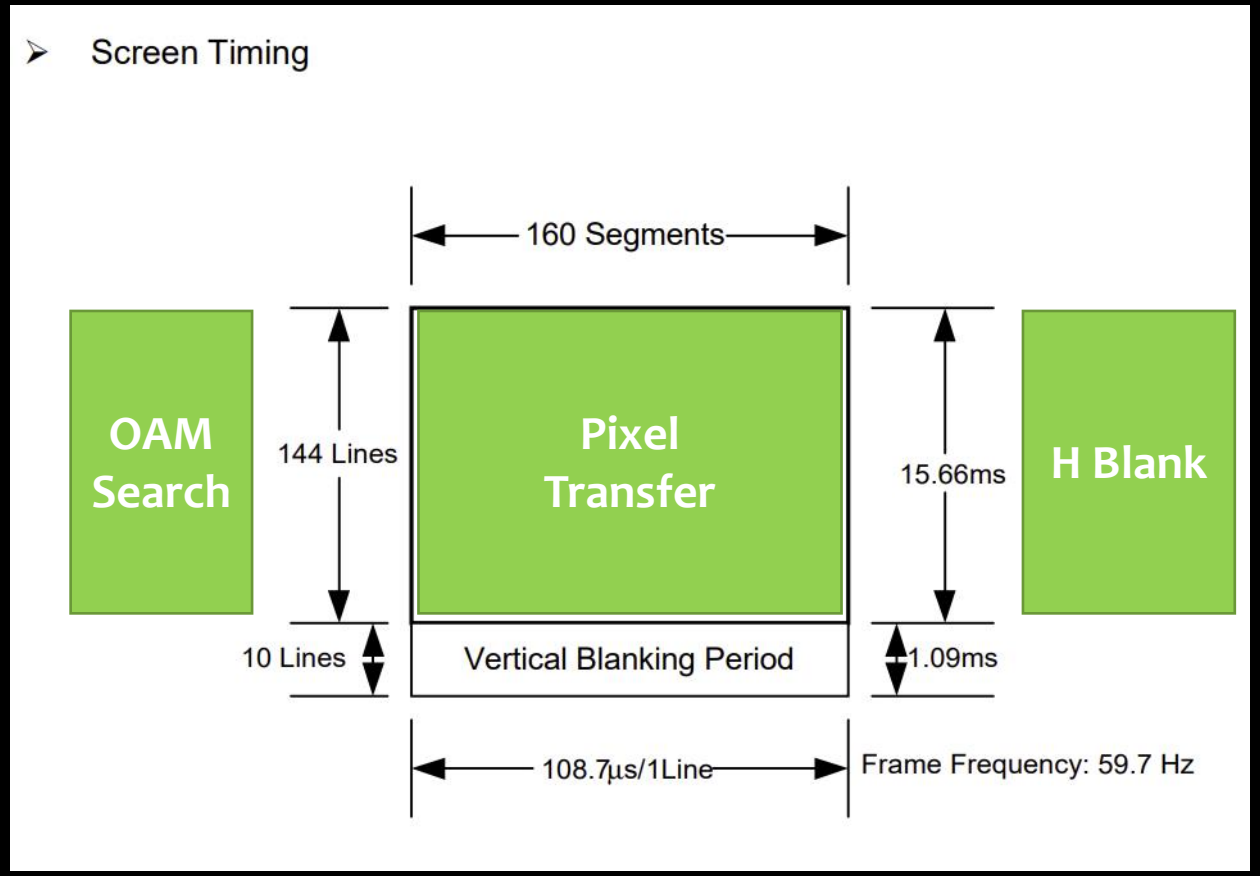
Next: ~~graphics~~ timing



Next: graphics & timing



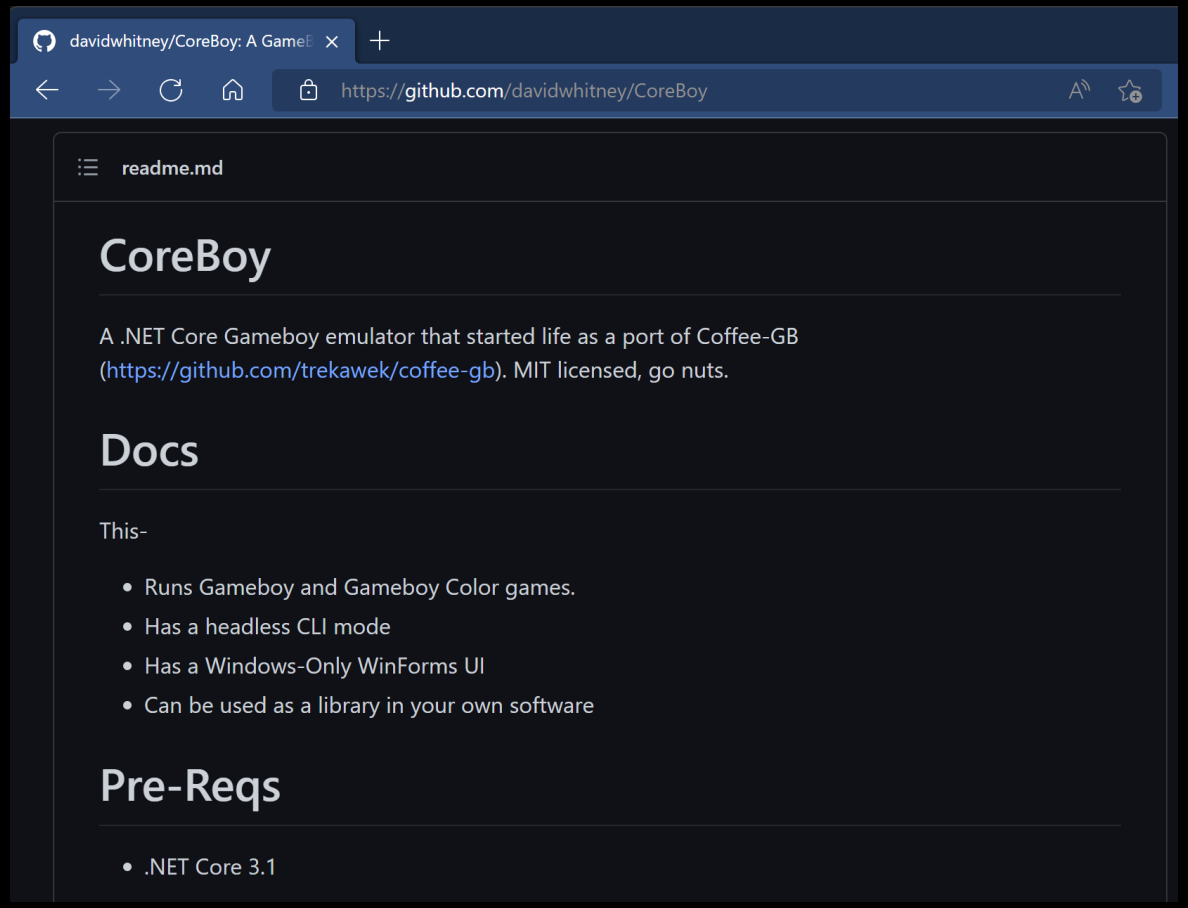
Next: graphics & timing



Time to look for a solution

And no, it wasn’t found on Stack Overflow

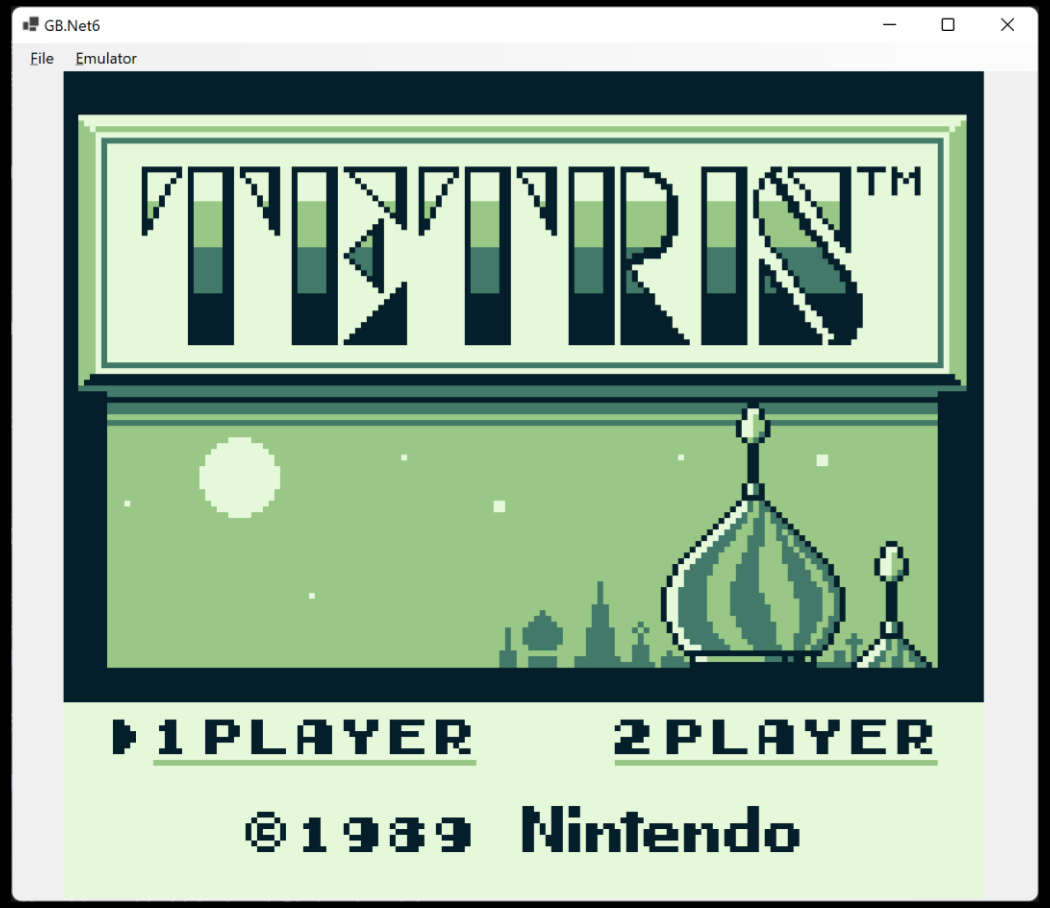
David Whitney to the rescue!



Upgrades

- .NET Core 3.1 => .NET 6
- Added sound support
- Fixed some sound channels
- Added custom UI control

Time to play some games!



Debugging

- Pressing START doesn’t do anything
  - Try another game first
  - START and all other buttons work just fine

Debugging

- Pressing START doesn’t do anything
  - Try another game first
  - START and all other buttons work just fine

- Breakpoint in the “CPU”
  - Stepping through every opcode?

Debugging

- Pressing START doesn't do anything
  - Try another game first
  - START and all other buttons work just fine

- Breakpoint in the "CPU"
  - Stepping through every opcode?

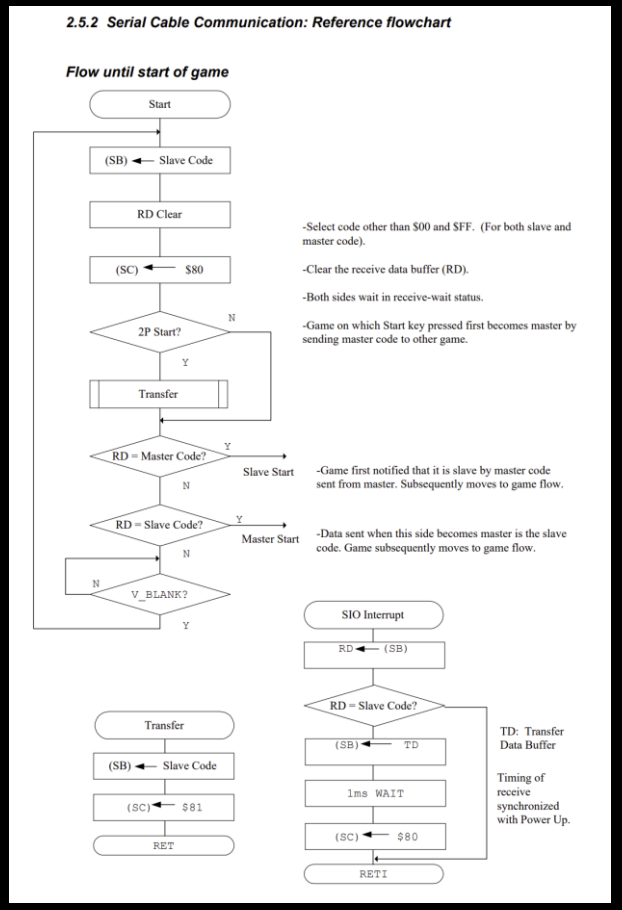
```
        LD SP,$fffe
        XOR A
        LD HL,$9fff
Addr_0007:
        LD (HL-),A
        BIT 7,H
        JR NZ, Addr_0007
```

Debugging

But wait, I know this game!

Eureka!

Tetris works!



2.5.2 Serial Cable Communication: Reference flowchart

Flow until start of game

Start

(SB) ← Slave Code

RD Clear

(SC) ← $80

2P Start? — N / Y

Transfer

RD = Master Code? — Y → Slave Start / N

RD = Slave Code? — Y → Master Start / N

V_BLANK? — N / Y

-Select code other than $00 and $FF. (For both slave and master code).

-Clear the receive data buffer (RD).

-Both sides wait in receive-wait status.

-Game on which Start key pressed first becomes master by sending master code to other game.

-Game first notified that it is slave by master code sent from master. Subsequently moves to game flow.

-Data sent when this side becomes master is the slave code. Game subsequently moves to game flow.

Transfer

(SB) ← Slave Code

(SC) ← $81

RET

SIO Interrupt

RD ← (SB)

RD = Slave Code?

(SB) ← TD

1ms WAIT

(SC) ← $80

RETI

TD: Transfer Data Buffer

Timing of receive synchronized with Power Up.

What’s next?

What’s next?

What's next?

- Game Boy Color support
  - Including running Game Boy games in GBC mode
- Making sure all test ROMs run successfully
  - Yes, there are "unit tests" for the Game Boy 😎
  - https://github.com/Gekkio/mooneye-gb
  - https://github.com/retrio/gb-test-roms

What’s next?

Time to look at the emulator!

One more thing…

One more thing…

GAME OVER