HW / SW Implementation Overview (0A)

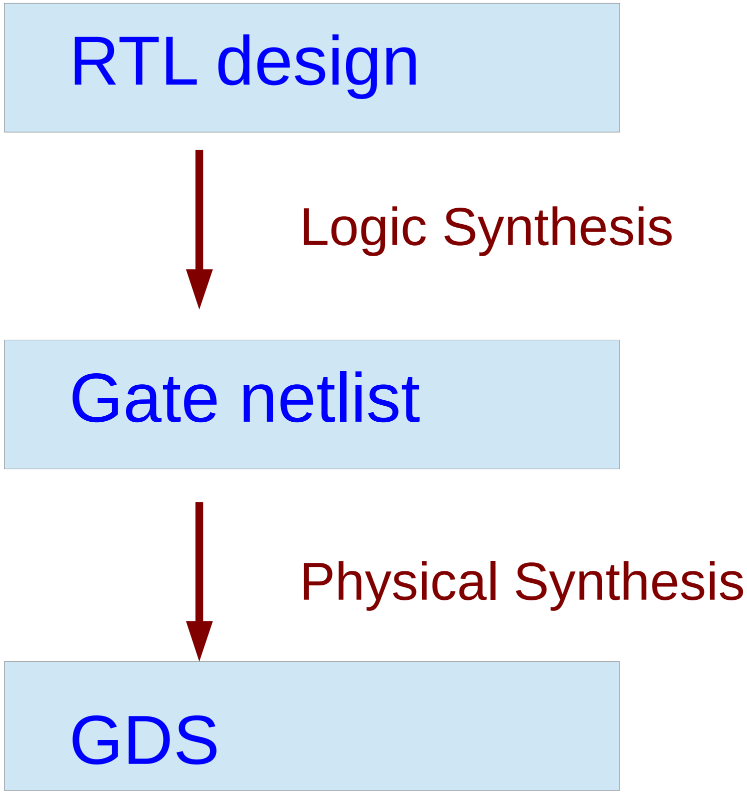
Young Won Lim
7/16/16

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

HW Implementation

ASIC (Application Specific Integrated Circuit)



RTL design

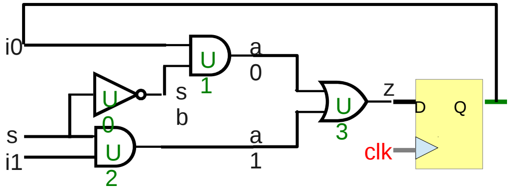
Logic Synthesis

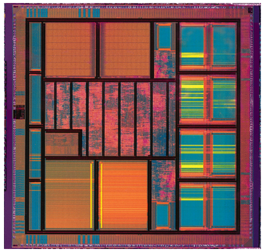
Gate netlist

Physical Synthesis

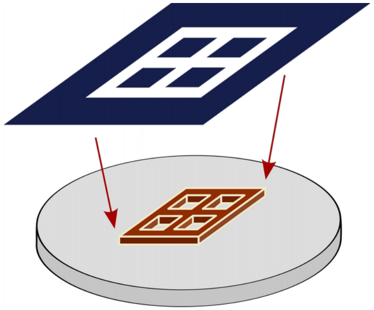
GDS

```
always @(posedge clk)
  if (load) q = d;
```



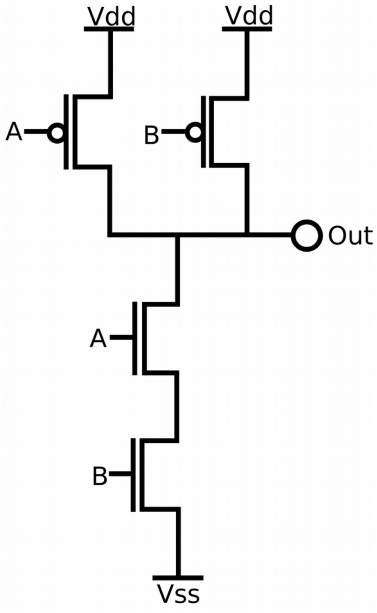


mask

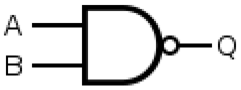


NAND Gate

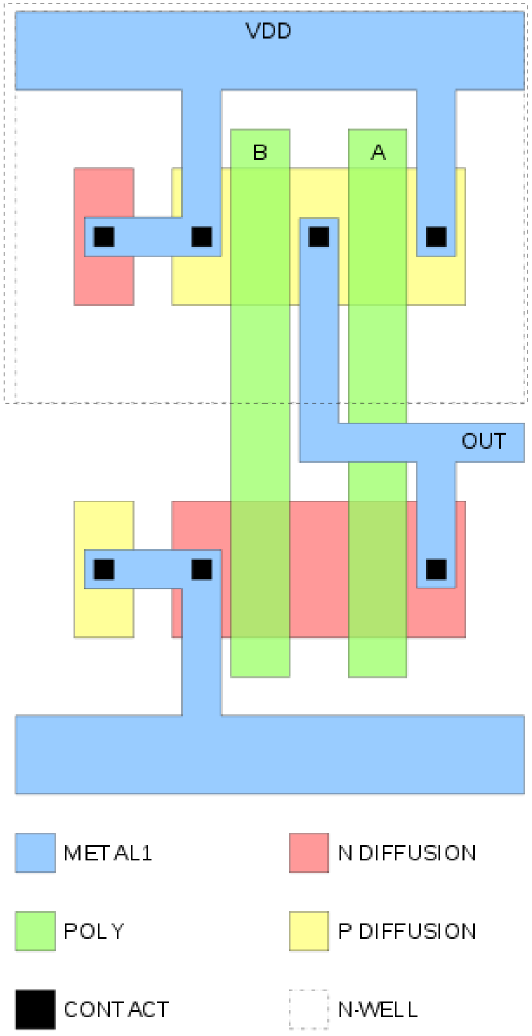
A Transistor Level



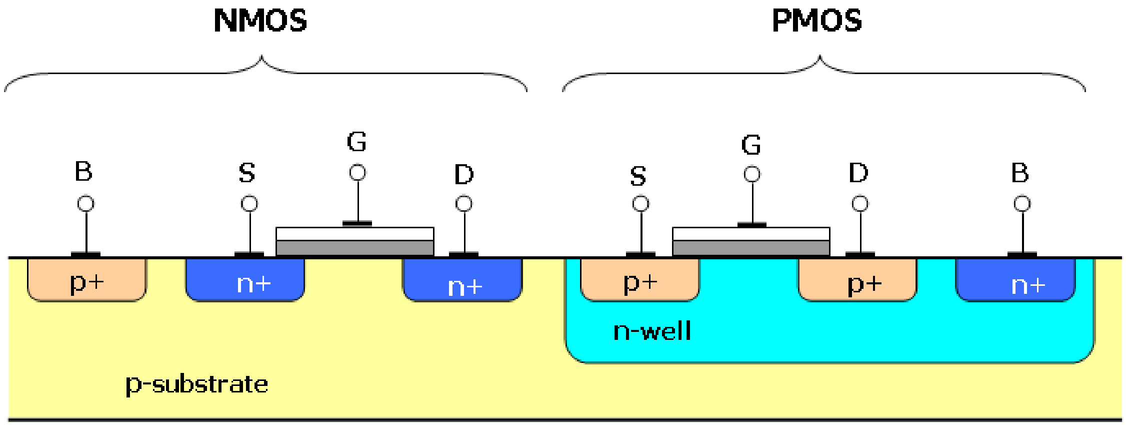
A Gate Level



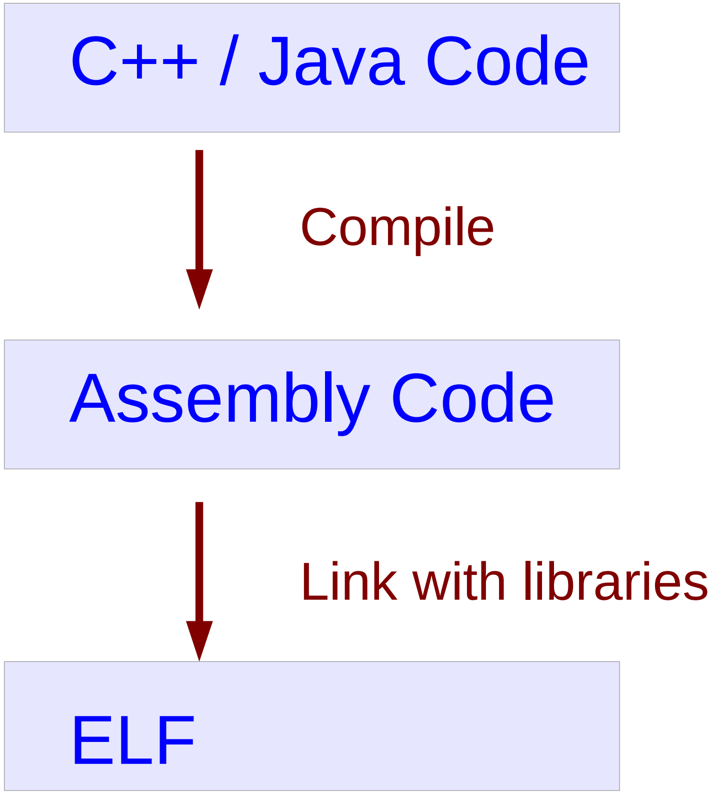
A Layout



A Wafer processed



SW Implementation



C++ / Java Code

Compile

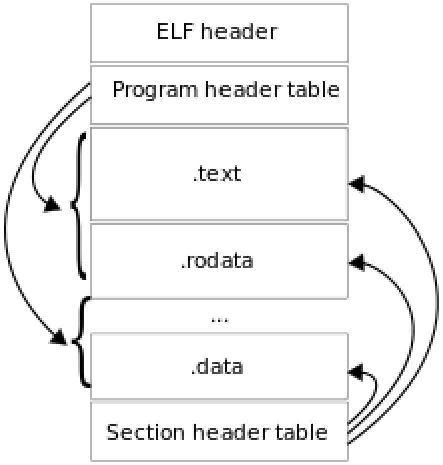
Assembly Code

Link with libraries

ELF

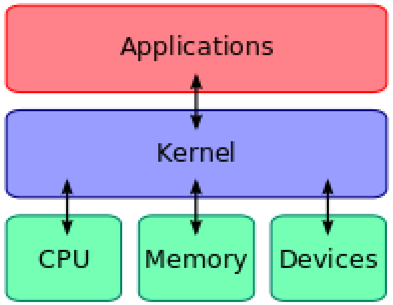
```
int i, s=0;
for(i=0; i<10; ++i)
 s+= i;
```

```
addi $s1, $0, 0
add  $s0, $0, $0
addi $t0, $0, 10
...
```



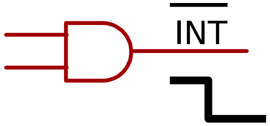
OS Kernel

- Process Management
- Multitasking
- Interrupts
- Modes
- Memory Management
- Virtual Memory
- Disk access
- File System
- Device Drivers

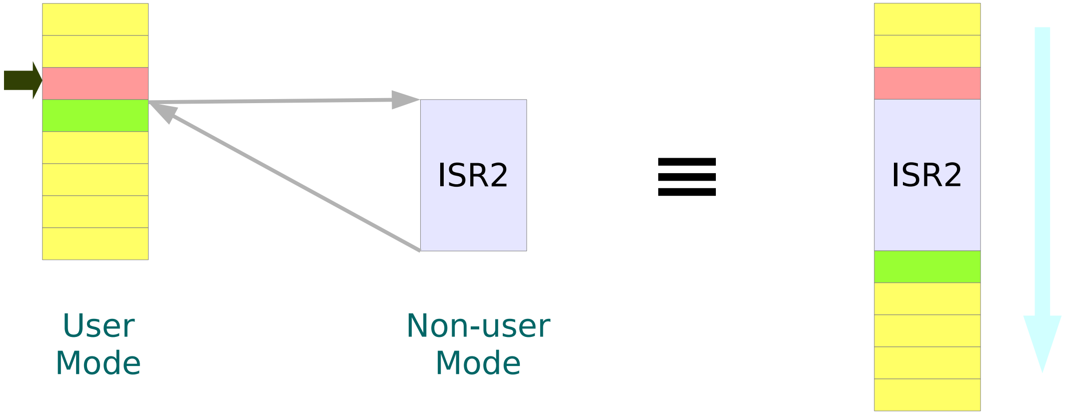


Interrupt

Asynchronous



Hardware Interrupt



User Mode

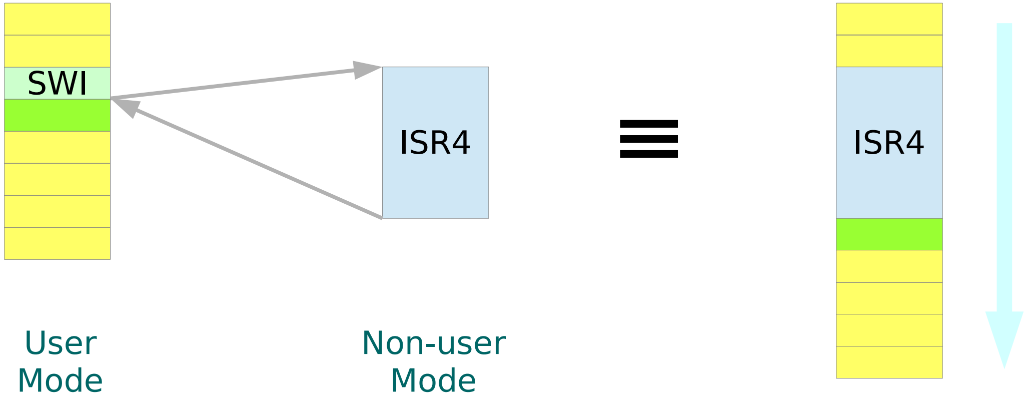
Non-user Mode

Synchronous

Software Interrupt Instruction

INT 0x80...

Software Interrupt

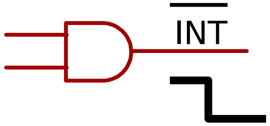


User Mode

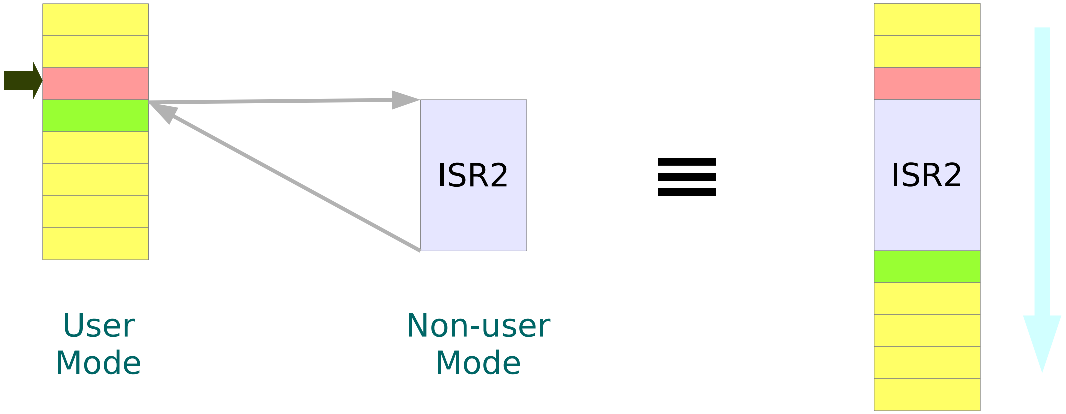
Non-user Mode

Interrupt and Parallelism

Asynchronous



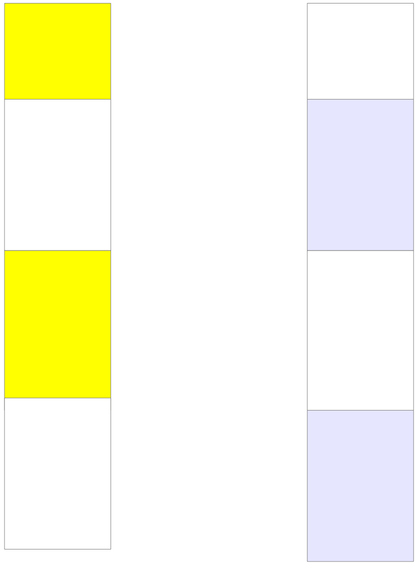
Hardware Interrupt



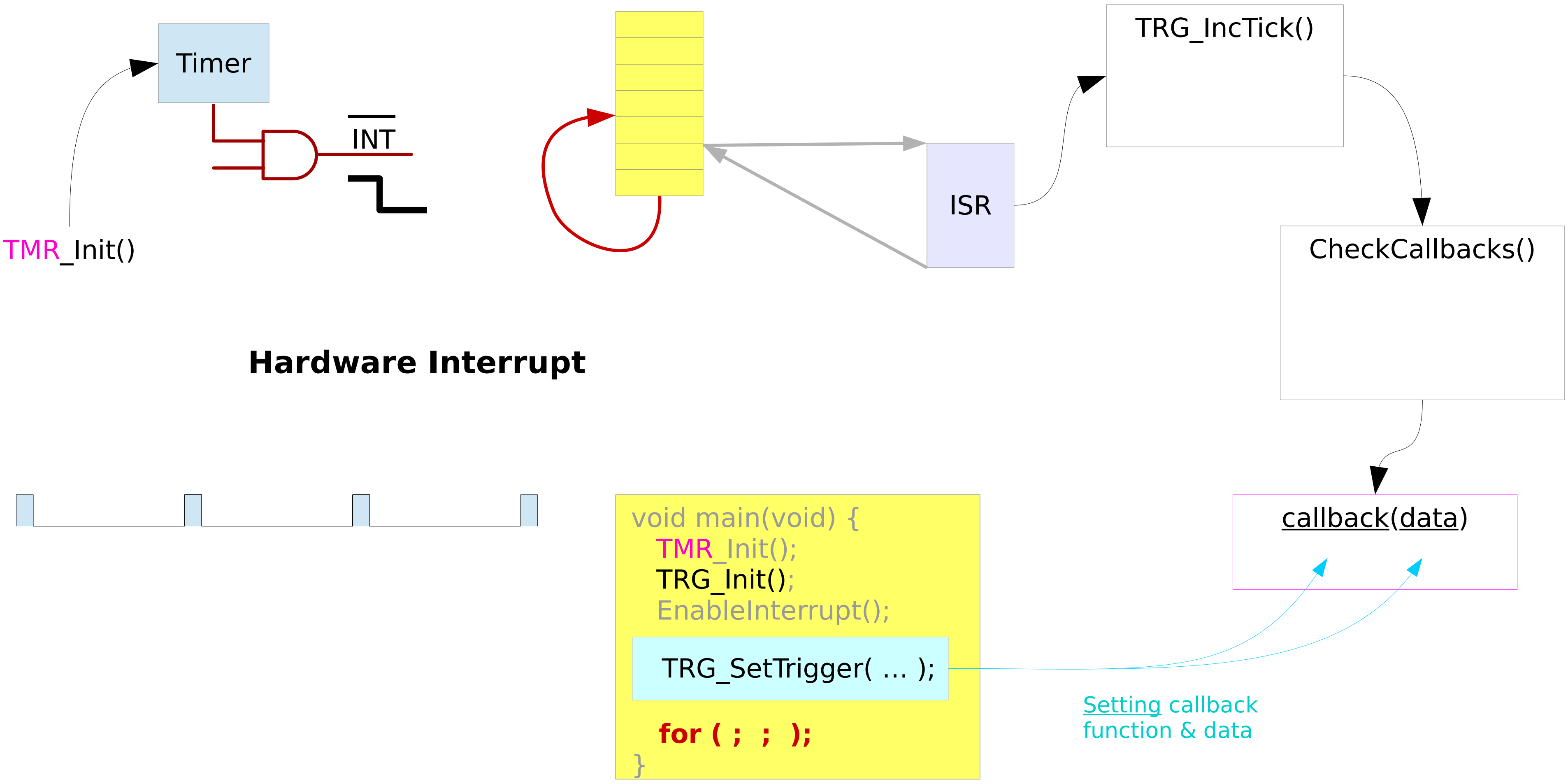
Two parallel threads of execution

Simulate parallelism

Time sharing



Timer & Interrupt



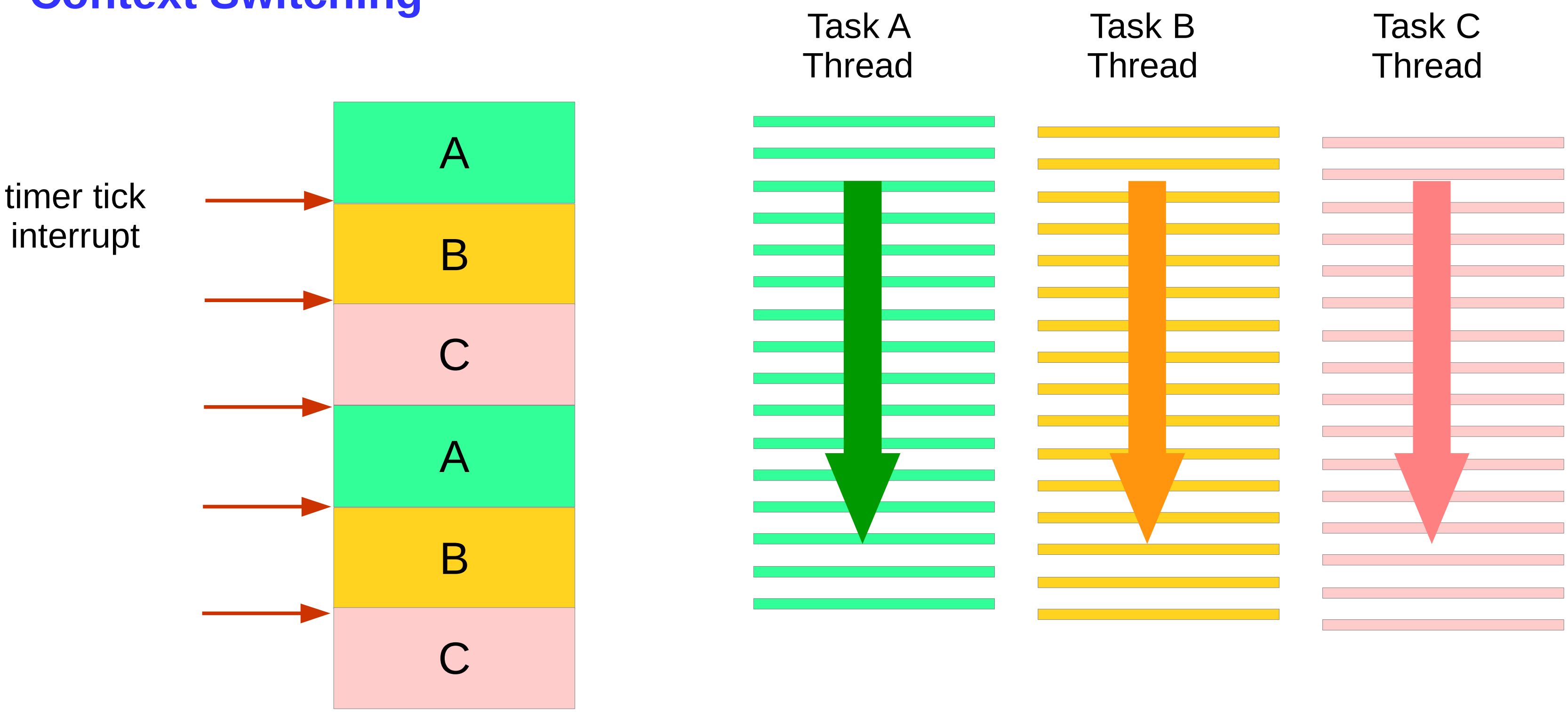
```
void main(void) {
   TMR_Init();
   TRG_Init();
   EnableInterrupt();

    TRG_SetTrigger( … );

   for ( ;  ;  );
}
```

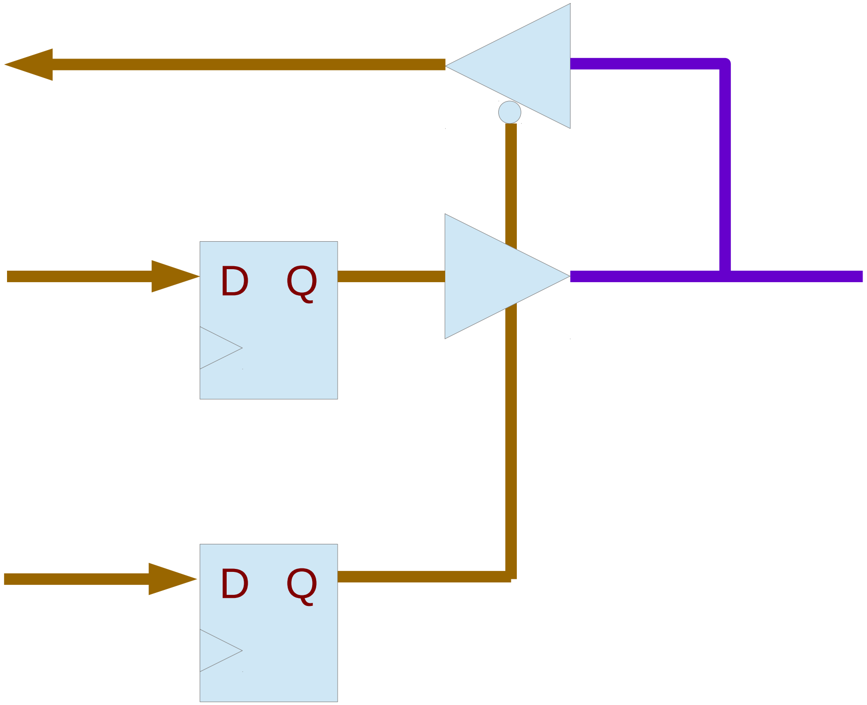
Multi-tasking

- **Time Slice**
- **Time Sharing**
- **Round Robin Scheduling**
- **Context Switching**



GPIO

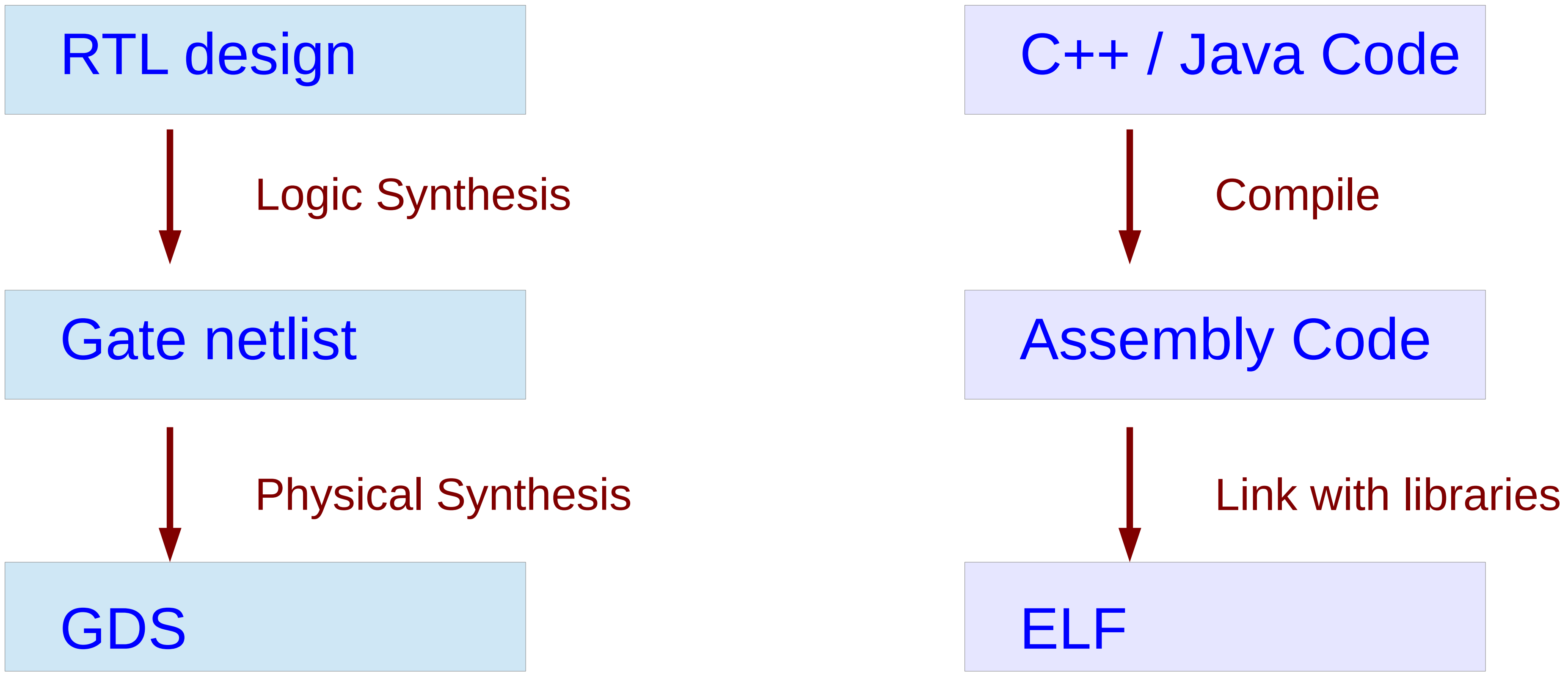
- read from port address
- write to port address
- write to port direction register



HW (ASIC) & SW Implementations

HW (ASIC)

SW



RTL design

Logic Synthesis

Gate netlist

Physical Synthesis

GDS

C++ / Java Code

Compile

Assembly Code

Link with libraries

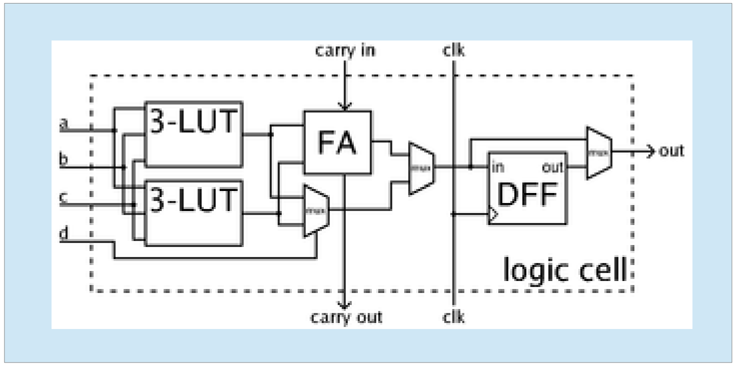
ELF

Parallelism is ensured by spatial allotment in a chip

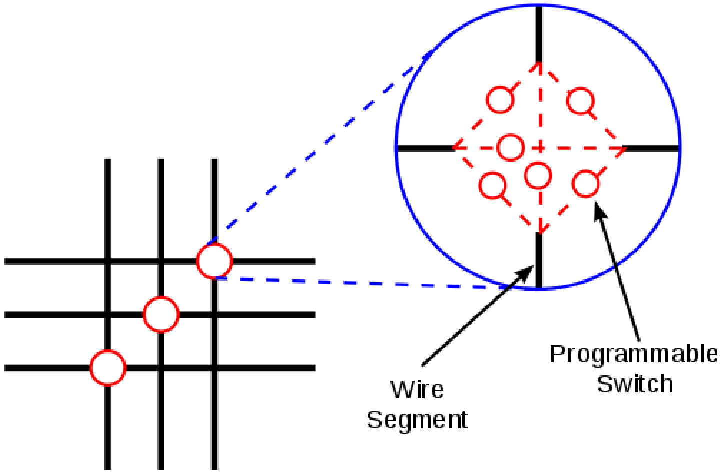
Parallelism is ensured by temporal processing on a processor

FPGA Architecture

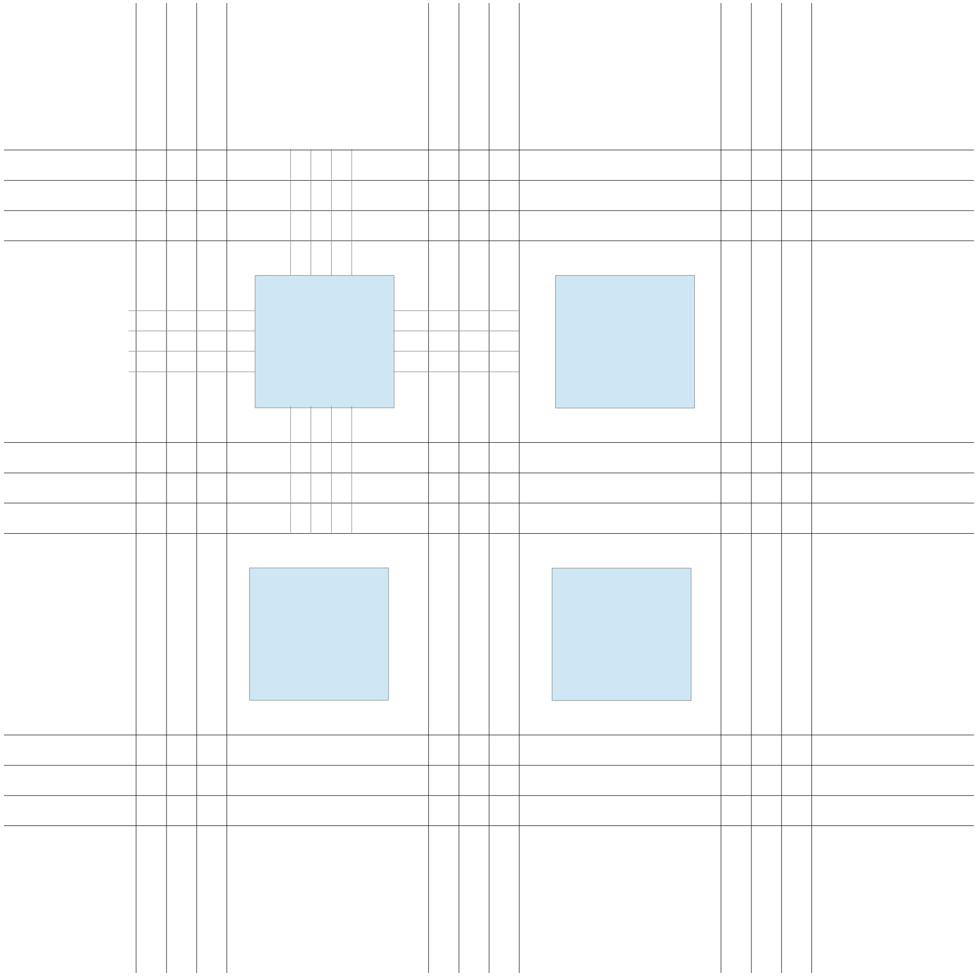
A Logic Block



A Configurable Interconnection



FPGA (Field Programmable Gate Array)



Implementing Boolean Functions

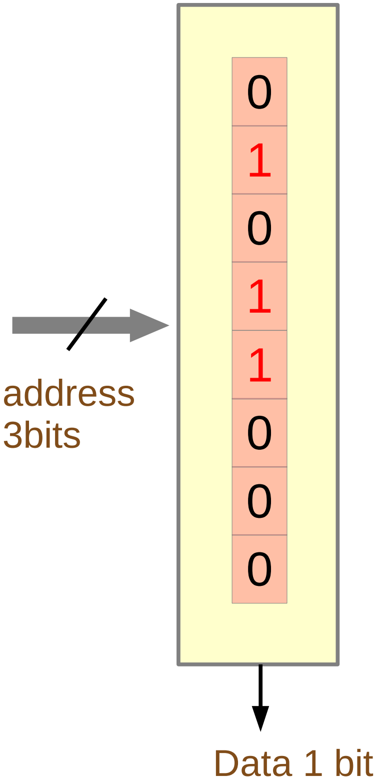
	x	y	z	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

index

inputs output

A Truth Table

Memory Size : 2^3 x 1



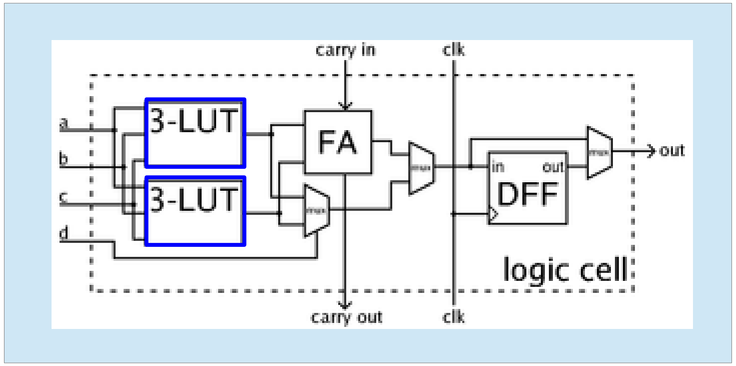
address 3bits

0
1
0
1
1
0
0
0

Data 1 bit

LUT (Look up table)

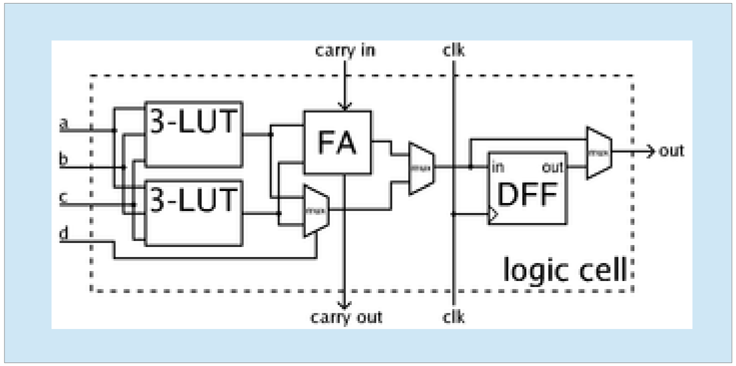
A Logic Block

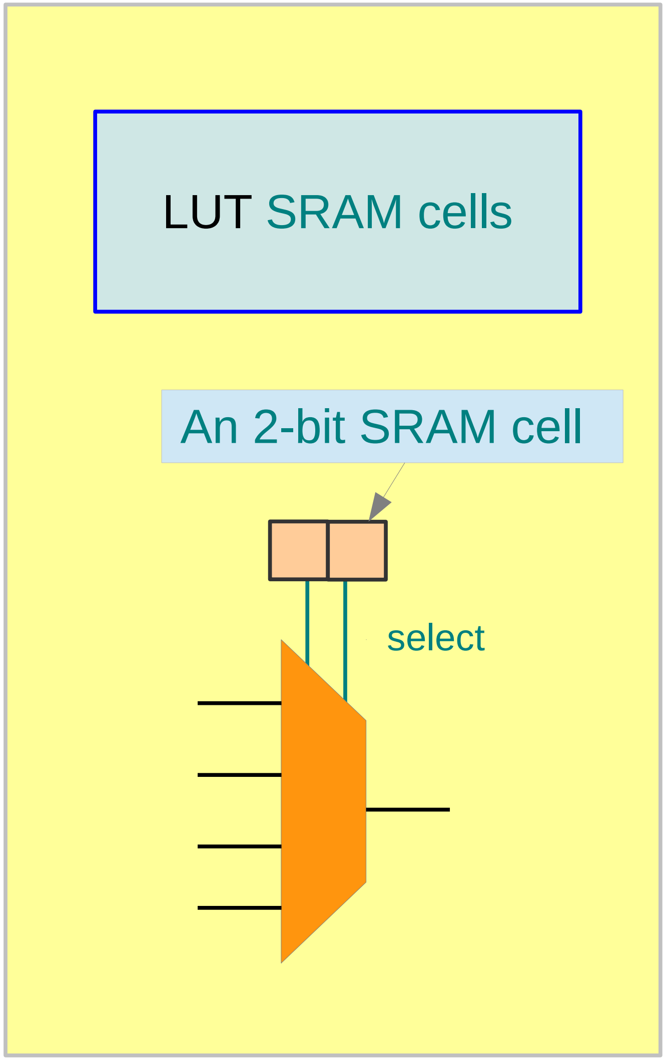


Combinational Logic

Configuring Logic Blocks

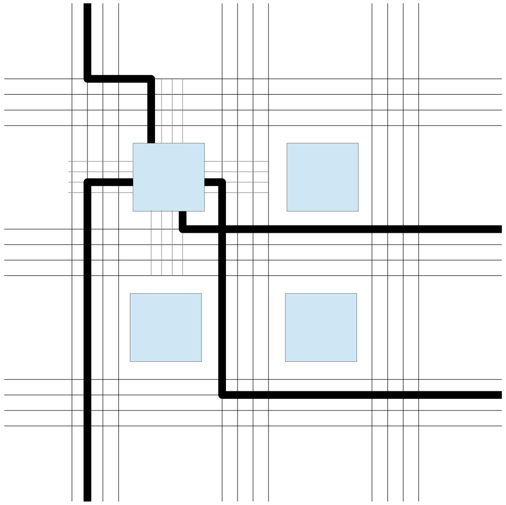
A Logic Block

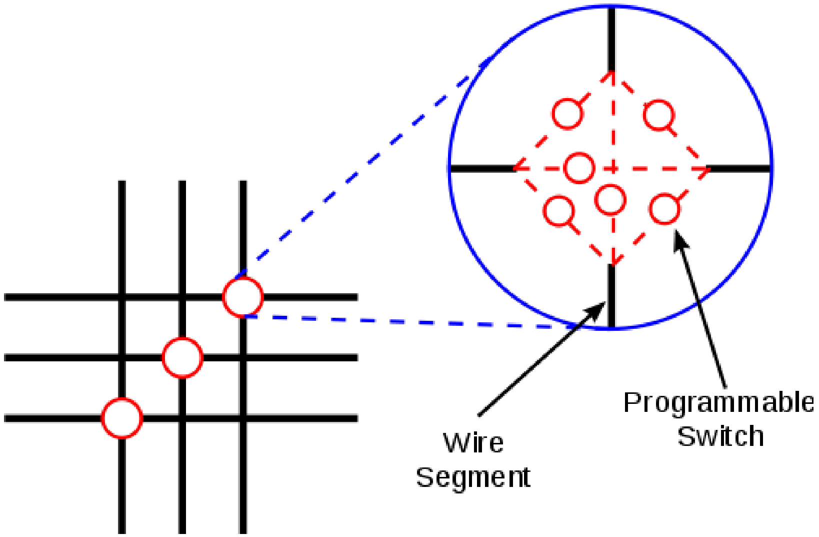


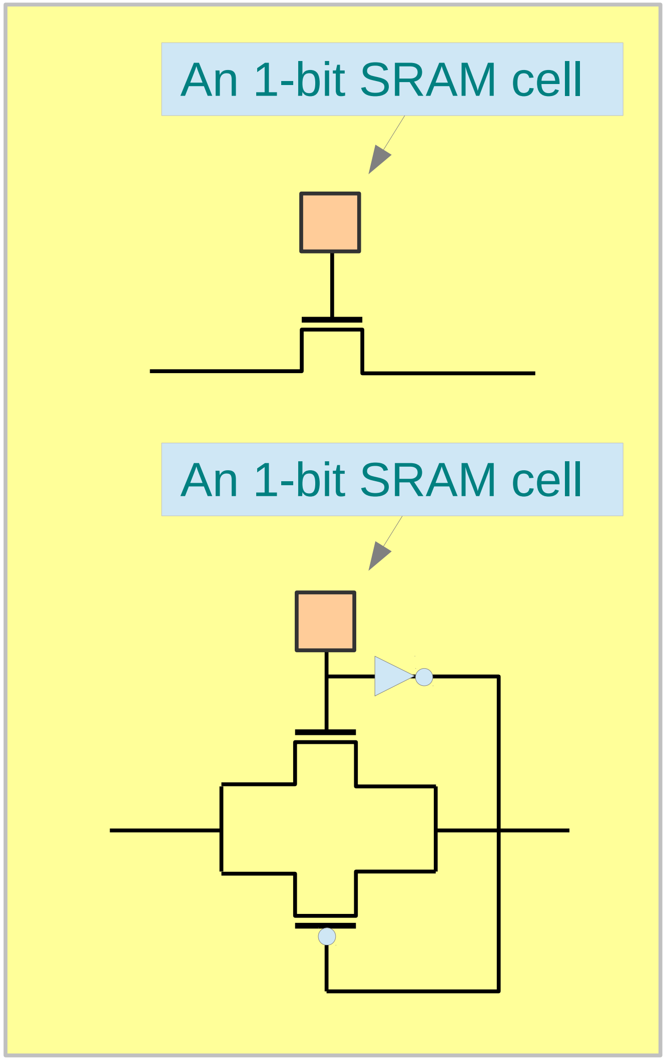

LUT SRAM cells

An 2-bit SRAM cell

Configuring Interconnections







An 1-bit SRAM cell

An 1-bit SRAM cell

HW (FPGA) Implementations

HW (FPGA)

RTL design

↓ Logic Synthesis

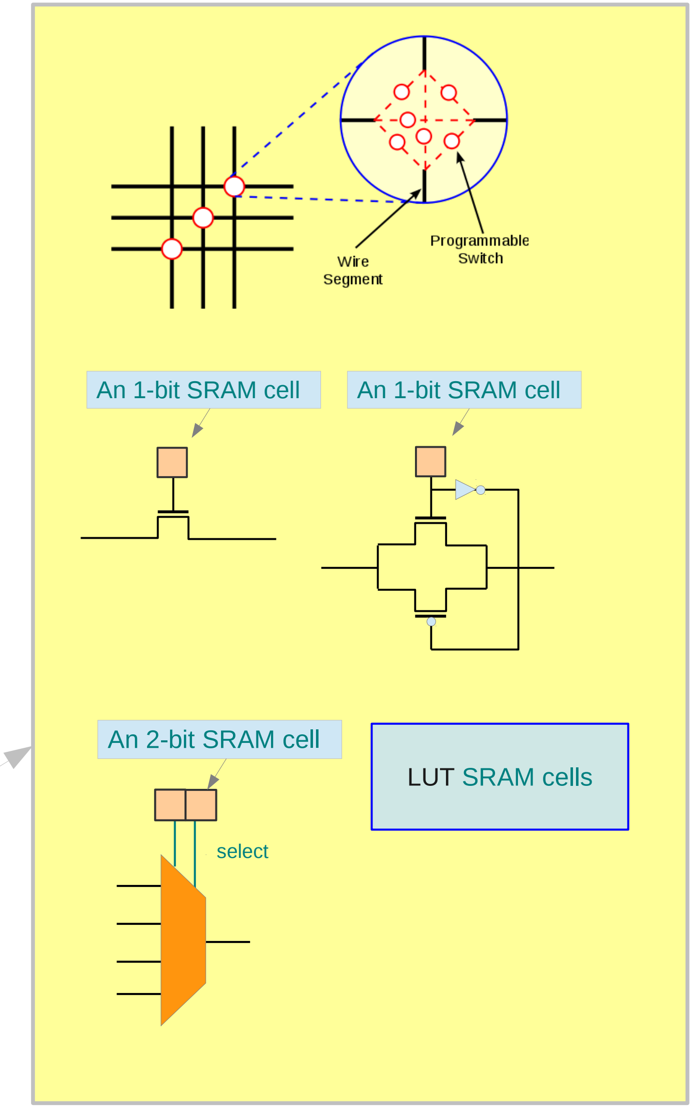
Gate netlist *

LUTs, FF, etc

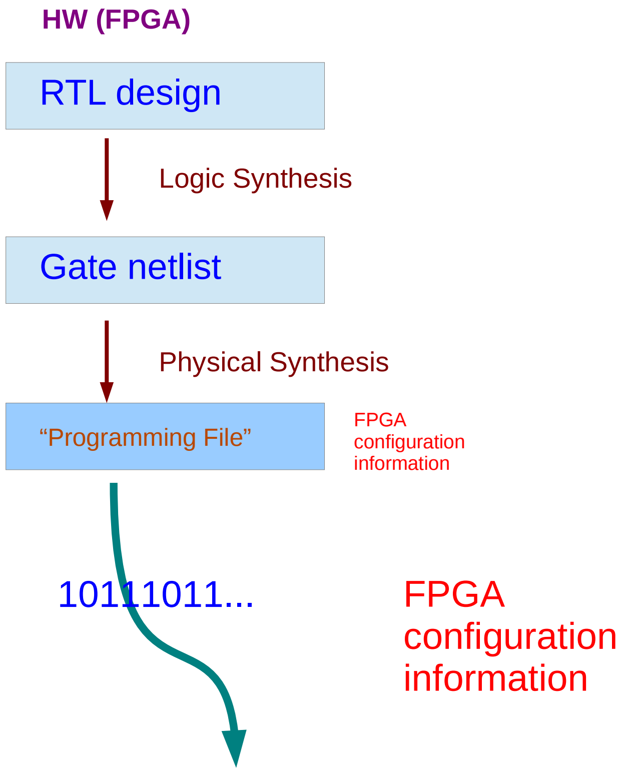
↓ Physical Synthesis

“Programming File”

FPGA configuration information



HW (FPGA) & SW Implementations



HW (FPGA)

RTL design

Logic Synthesis

Gate netlist

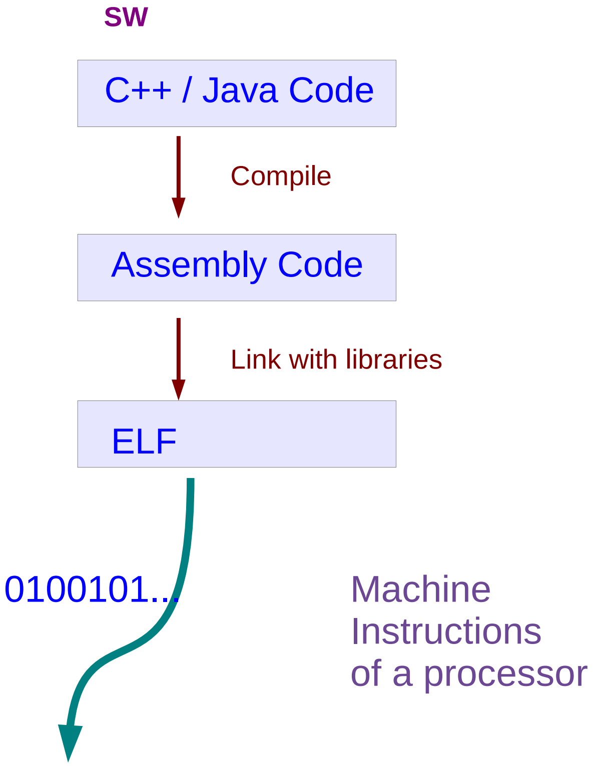
Physical Synthesis

“Programming File”

FPGA configuration information

10111011...

FPGA configuration information



SW

C++ / Java Code

Compile

Assembly Code

Link with libraries

ELF

0100101...

Machine Instructions of a processor

References

[1] http://en.wikipedia.org/
[2] D.M. Harris, S. L. Harris, “Digital Design and Computer Architecture”