Embedded System Development

System-on-Chip: Xilinx EDK

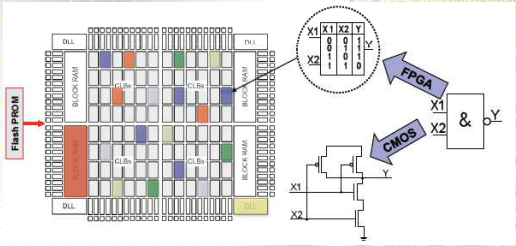
George Kornaros

Agenda

- FPGAs
- FPGA Tool Flow
- System on Chip (SoC)
- SoC Tool Flow
- Demonstration

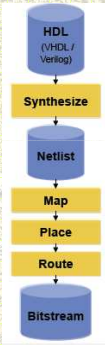
Field-Programmable Gate Arrays (FPGAs)

- Fine-grained reconfigurable hardware
- Gate-Array: regular structure of "logic cells", connected through an interconnection network
- Configuration stored in SRAM, must be loaded on startup

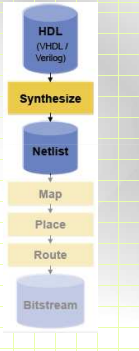


FPGA Tool Flow

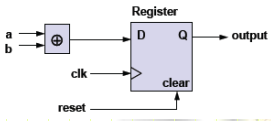
- Hardware design is traditionally done by modeling the system in a hardware description language
- An FPGA "compiler" (synthesis tool) generates a netlist,
- which is then mapped to the FPGA technology,
- the inferred components are placed on the chip,
- and the connecting signals are routed through the interconnection network.



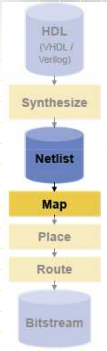
HDL Synthesis

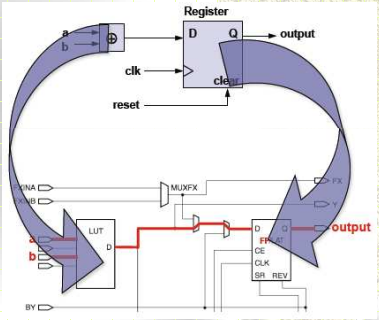



```
process(clk, reset)
begin
  if reset = '1' then
    output <= '0';
  elsif clk'event AND clk = '1' then
    output <= a XOR b;
  end if;
end process;
```

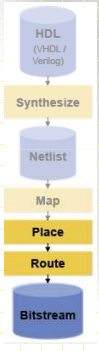


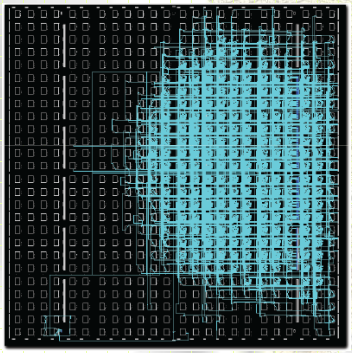
Technology Mapping



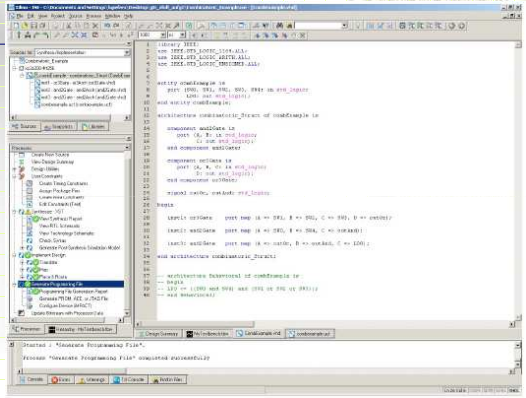


Place and Route

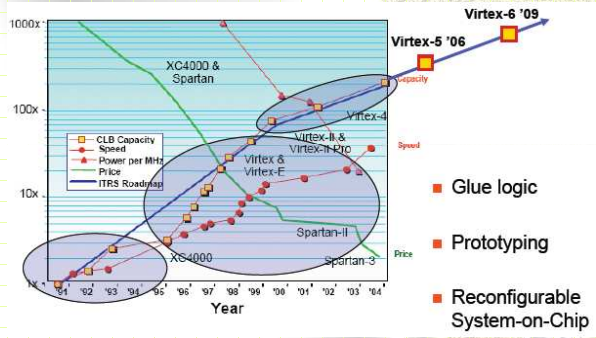




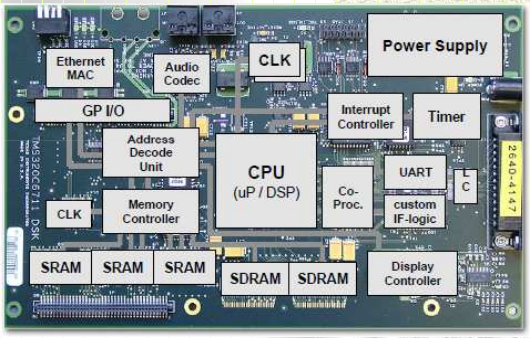
Xilinx ISE



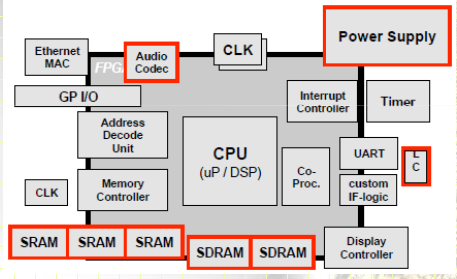
FPGA Evolution



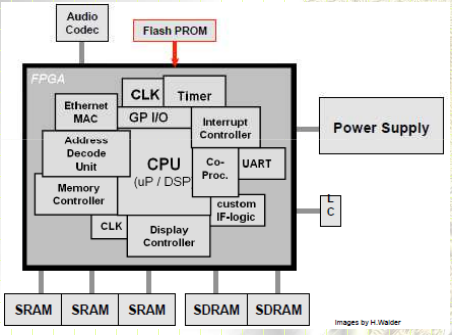
Traditional Embedded System



Embedded System Components



Configurable System-on-Chip

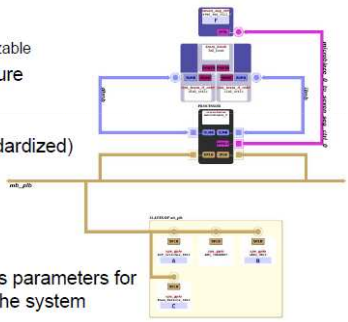


Advantages

- Fewer physical components
- Shorter development cycles
- Field-programmable (updates, new features...)
  - ✓ Custom co-processors / accelerators
- Possibly higher performance through on-chip integration
  - ✓ Signals on a chip can typically be clocked higher than signals across board traces
  - ✓ Optimization between modules possible
- Partial reconfigurability
  - ✓ Exchange peripherals while the rest of the system keeps running

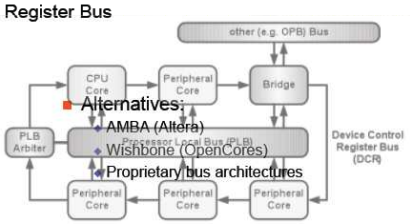
Building a SoC

- HW-Blocks are called IP "cores"
  - Buy them like Software
  - Implement own cores
  - Usually highly parametrizable
- Interconnect infrastructure
  - Bus Architectures
  - P2P Links
- Cores implement (standardized) interfaces
- Modelling kit determines parameters for cores and "generates" the system



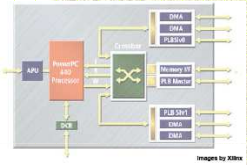
Bus Architecture: CoreConnect

- Flexible bus architecture for embedded Systems and SoCs
  - Developed by IBM
  - Used by Xilinx EDK
- Processor Local Bus (PLB)
- On-Chip Peripheral Bus (OPB)
- Device Control Register Bus (DCR)
- Alternatives:
  - AMBA (Altera)
  - Wishbone (OpenCores)
  - Proprietary bus architectures



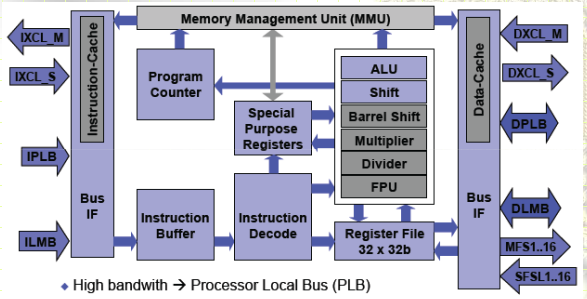
Embedded CPUs

- PowerPC 440 (hard core)
  - 32 bit embedded PowerPC RISC
  - Up to 550 MHz
  - 2x 32 kB instruction and data caches
  - Memory management unit (MMU)
  - Hardware multiply and divide
  - Coprocessor interface (APU)
  - PLB and DCR bus interfaces



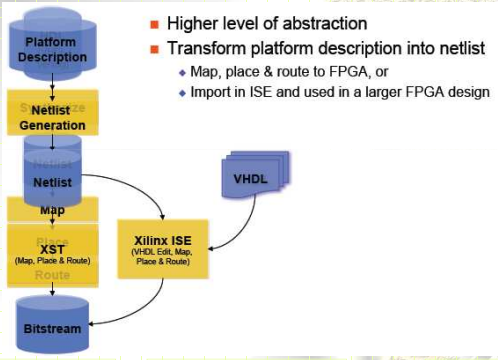
- MicroBlaze (soft core)
  - 32 bit RISC architecture, up to 235 MHz
  - Min. configuration: 1,010 LUTs [XC5VLX330 FPGA: 207,360 LUTs ➔ .5%]
  - Highly configurable
    - Barrel Shifter
    - MMU, 2-64 kB instruction and data caches
    - HW Multiplier/Divider, FPU, Debug
  - PLB, LMB, FSL bus interfaces
- Others
  - NIOS (Altera)
  - ARM
  - PicoBlaze (Xilinx), ...

MicroBlaze Configuration

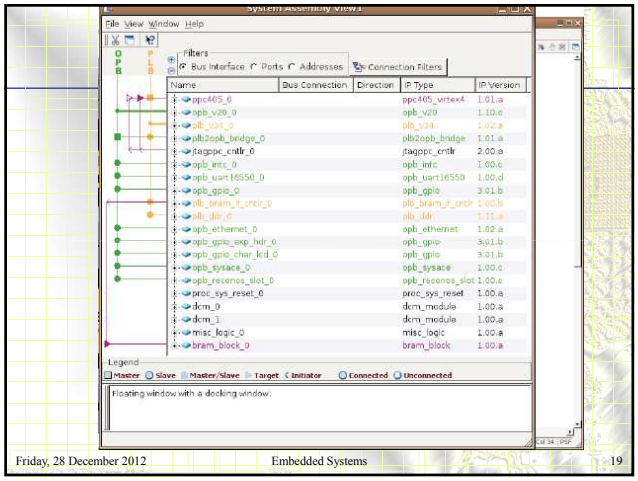


- High bandwith → Processor Local Bus (PLB)
- Configurable caches → Xilinx Cache Link (XCL)
- Local (BRAM) memory → Local Memory Bus (LMB)
- Co-processor port → Fast Simplex Link (FSL)

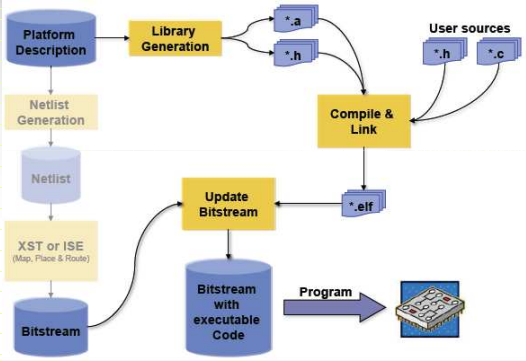
Design Flow (Hardware)

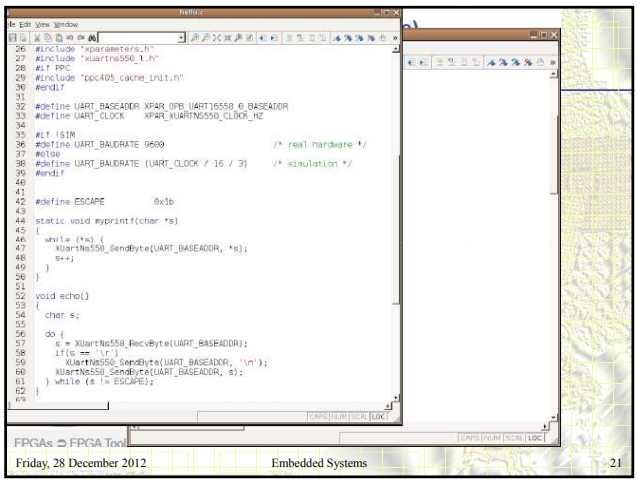


- Higher level of abstraction
- Transform platform description into netlist
  - Map, place & route to FPGA, or
  - Import in ISE and used in a larger FPGA design



Design Flow (Software)





```
#include "xparameters.h"
#include "xuartns550_l.h"
#if PPC
#include "ppc405_cache_init.h"
#endif

#define UART_BASEADDR XPAR_OPB_UART16550_0_BASEADDR
#define UART_CLOCK    XPAR_XUARTNS550_CLOCK_HZ

#if !SIM
#define UART_BAUDRATE 9600                  /* real hardware */
#else
#define UART_BAUDRATE (UART_CLOCK / 16 / 3)  /* simulation */
#endif

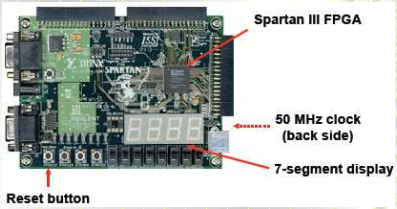
#define ESCAPE          0x1b

static void myprintf(char *s)
{
  while (*s) {
    XUartNs550_SendByte(UART_BASEADDR, *s);
    s++;
  }
}

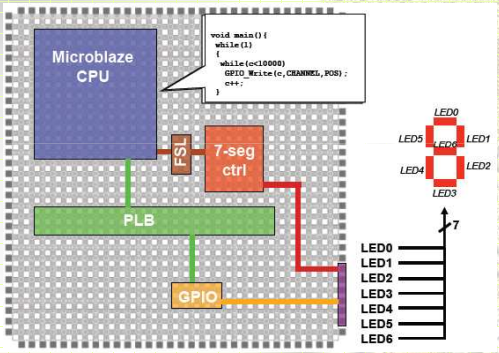
void echo()
{
  char s;

  do {
    s = XUartNs550_RecvByte(UART_BASEADDR);
    if(s == '\r')
      XUartNs550_SendByte(UART_BASEADDR, '\n');
    XUartNs550_SendByte(UART_BASEADDR, s);
  } while (s != ESCAPE);
}
```


Demonstration



Spartan-3 Demonstration



Questions



