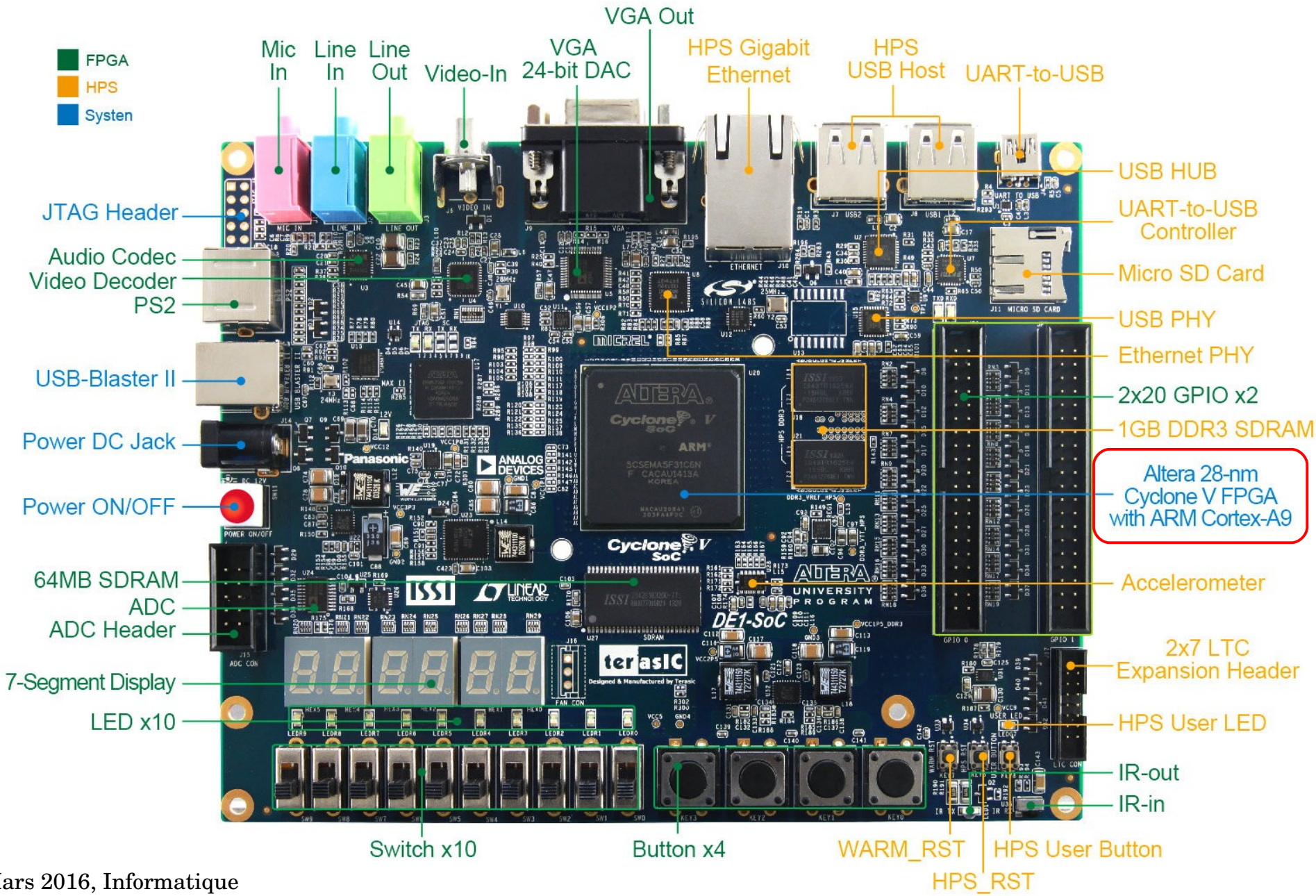
FPGA: accélérateur de calcul

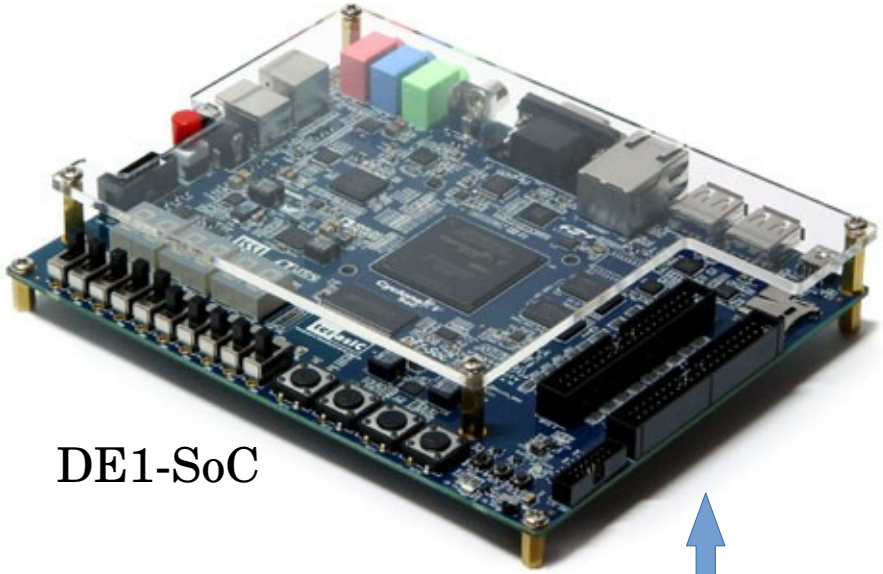
Pôle ITT, 08.03.2017

Bogdan Vulpescu

DE1-SoC: Cyclone V FPGA, ARM Cortex-A9



Linux sur carte MicroSD:
- avec pont vers FPGA [1]
- avec un module dans le kernel (pilote linux)



DE1-SoC

UART

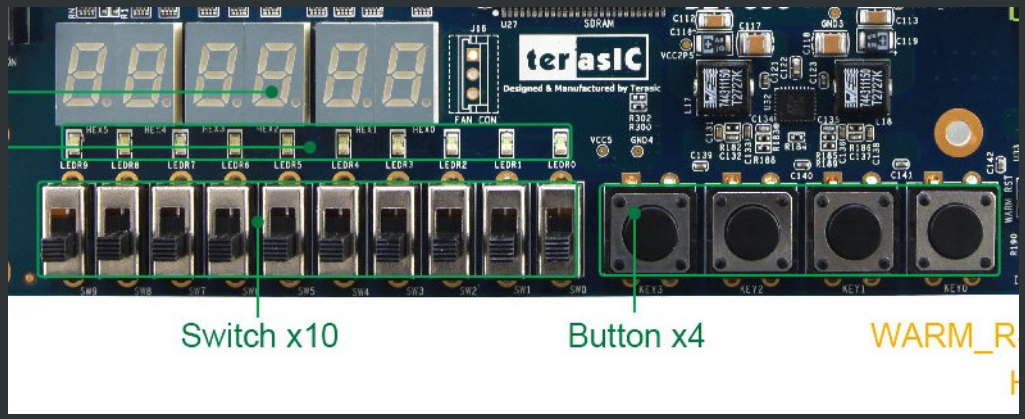
ARM Linux image*


```
EXT4-fs (mmcblk0p2): mounted filesystem with ordered data mode. Opts: (null)
VFS: Mounted root (ext4 filesystem) on device 179:2.
devtmpfs: mounted
Freeing unused kernel memory: 364K (806c8000 - 80723000)
init: ureadahead main process (51) terminated with status 5

Last login: Thu Jan  1 00:00:09 UTC 1970 on tty1
root@de1soclinux:~#
root@de1soclinux:~#
root@de1soclinux:~#
root@de1soclinux:~#
root@de1soclinux:~# ls -l
total 5964
-rwxr----- 1 root root 3742980 Jan  1  1970 DE1_SoC_Computer.rbf
drwxr-xr-x 6 root root    4096 Jan  1  1970 OpenCL_Examples
drwxr-xr-x 2 root root    4096 Jan  1  1970 animation_bouncy
drwxr-xr-x 2 root root    4096 Jan  1  1970 drawbox
drwxr-xr-x 2 root root    4096 Jan  1  1970 gsensor
drwxr-xr-x 2 root root    4096 Jan  1  1970 helloworld
drwxr-xr-x 2 root root    4096 Jan  1  1970 increment_leds
-rw-r--r-- 1 root root     279 Jul  2  2015 init_opencl.sh
drwxr-xr-x 2 root root    4096 Jan  1  1970 leds
drwxr-xr-x 2 root root    4096 Jan  1  1970 mem_rw
-rwxr-xr-x 1 root root 2316870 Jan  1 00:00 opencl.rbf
drwxr-xr-x 7 root root    4096 Jan  1  1970 opencl_arm32_rte
drwxr-xr-x 3 root root    4096 Jan  1  1970 pushbutton_irq_handler
root@de1soclinux:~#
CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7 | VT102 | Offline | ttyUSB0
```

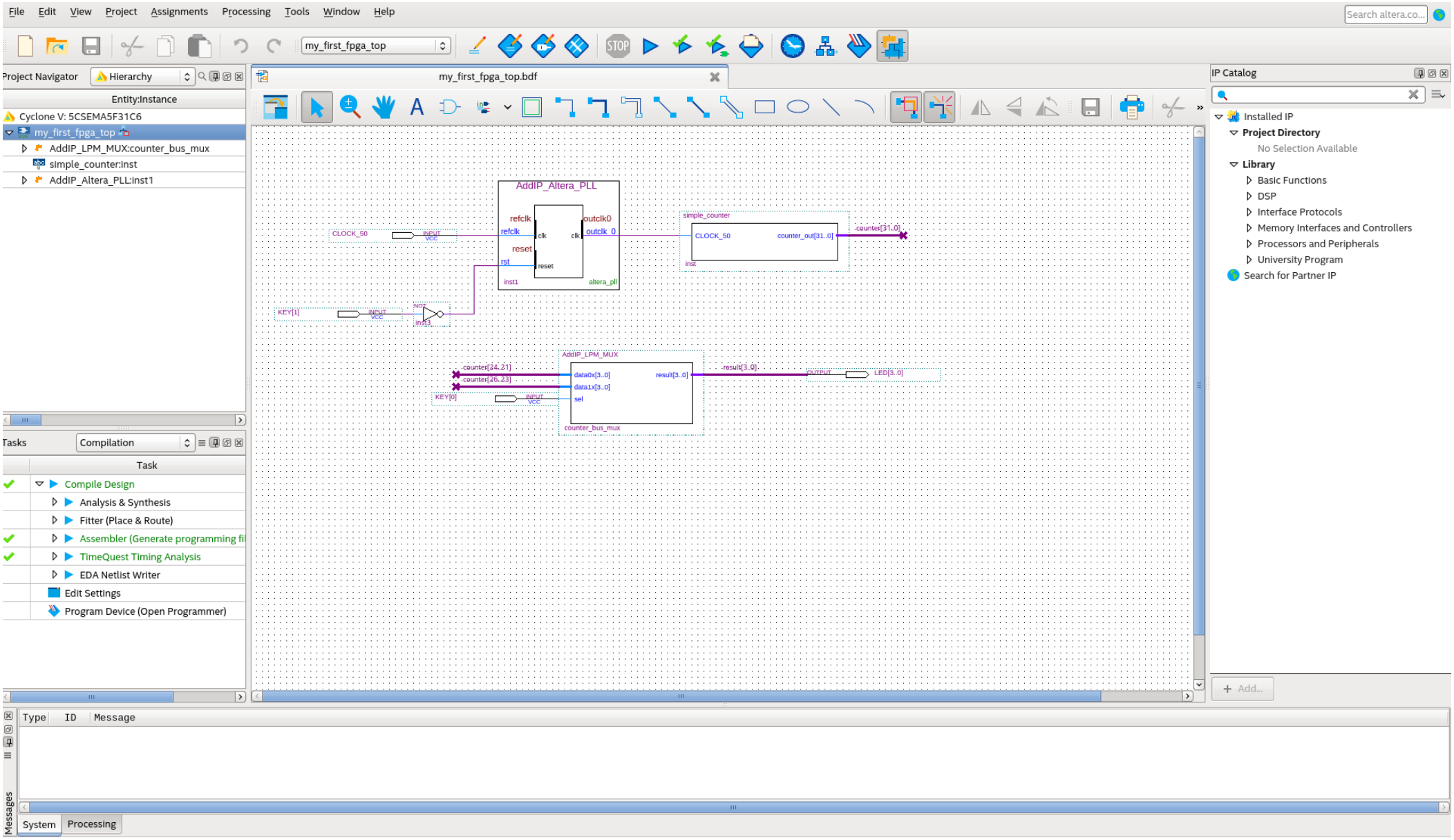

[1] https://people.ece.cornell.edu/land/courses/ece5760/DE1_SOC/HPS_peripherials/index.html

HPS-FPGA bridge exemple “increment_leds.c”

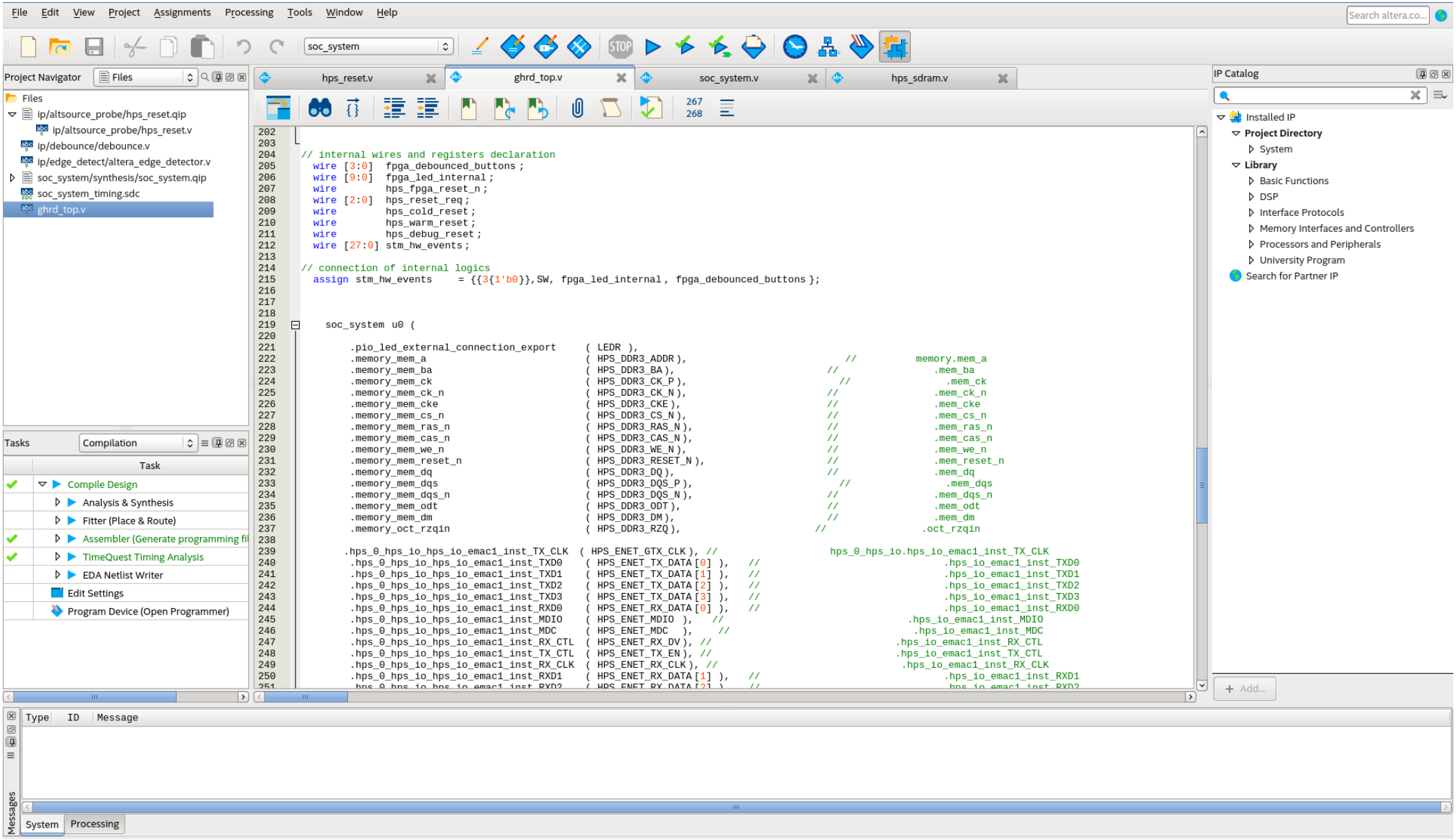
```c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/mman.h>

#define HW_REGS_BASE ( 0xff200000 )
#define HW_REGS_SPAN ( 0x00200000 )
#define HW_REGS_MASK ( HW_REGS_SPAN - 1 )
#define LED_PIO_BASE 0x0

int main(void)
{
    volatile unsigned int *h2p_lw_led_addr=NULL;
        void *virtual_base;
        int fd;

    // Open /dev/mem
        if( ( fd = open( "/dev/mem", ( O_RDWR | O_SYNC ) ) ) == -1 ) {
                printf( "ERROR: could not open \"/dev/mem\"...\n" );
                return( 1 );
        }

    // get virtual addr that maps to physical
        virtual_base = mmap( NULL, HW_REGS_SPAN, ( PROT_READ | PROT_WRITE ), MAP_SHARED, fd, HW_REGS_BASE );
        if( virtual_base == MAP_FAILED ) {
                printf( "ERROR: mmap() failed...\n" );
                close( fd );
                return(1);
        }

    // Get the address that maps to the LEDs
        h2p_lw_led_addr=(unsigned int *)(virtual_base + (( LED_PIO_BASE ) & ( HW_REGS_MASK ) ));

    // Add 1 to the PIO register
    *h2p_lw_led_addr = *h2p_lw_led_addr + 1;

        if( munmap( virtual_base, HW_REGS_SPAN ) != 0 ) {
                printf( "ERROR: munmap() failed...\n" );
                close( fd );
                return( 1 );

        }
        close( fd );
        return 0;
}
root@de1soclinux:~/increment_leds#
```



CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7 | VT102 | Offline | ttyUSB0

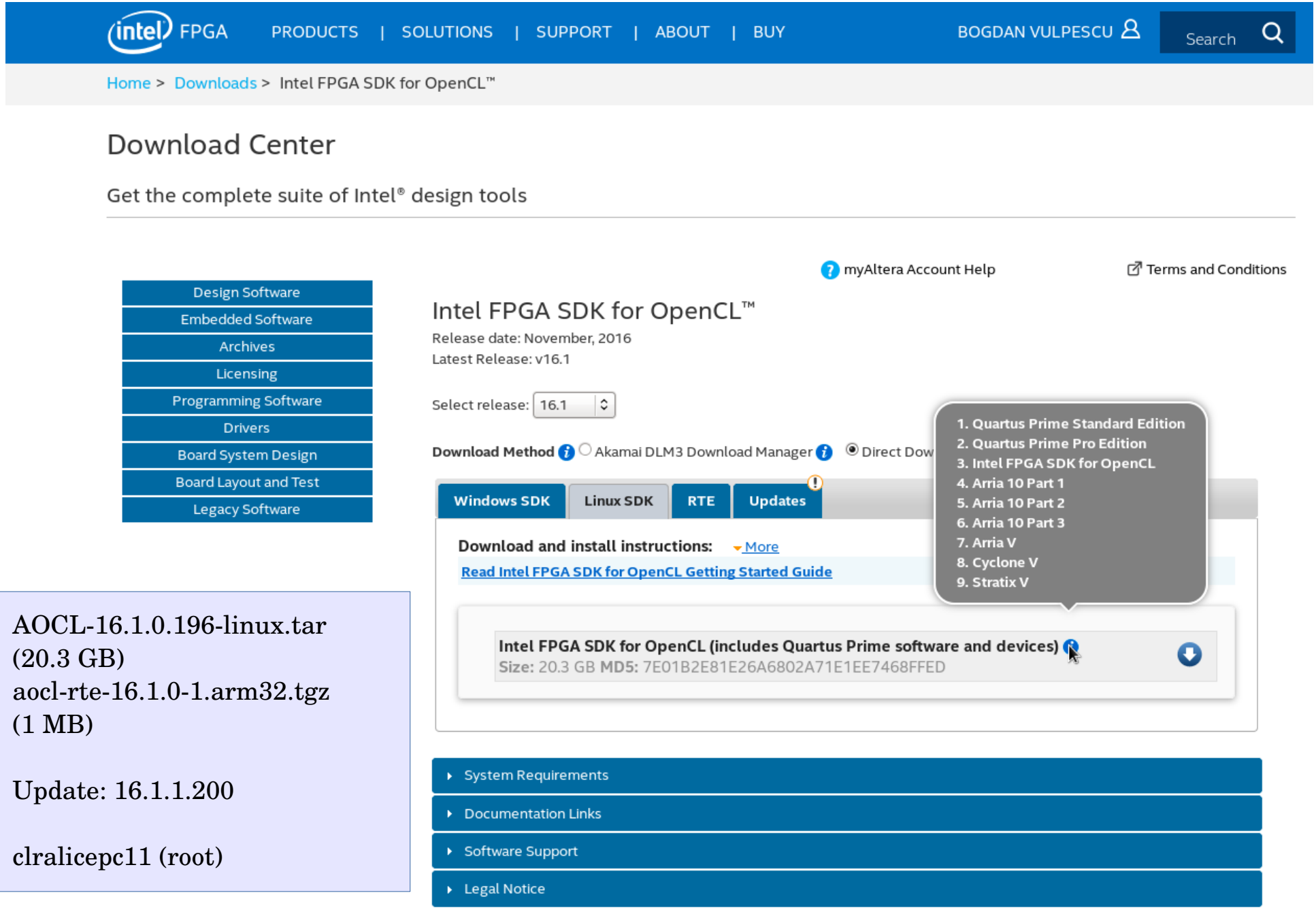


“my_first_fpga” - avec coeurs IP
- compiled SRAM Object File .sof (JTAG chain)



“my_first_hps-fpga_base” - description hardware Verilog)
- .sof sur FPGA file via JTAG chain
- ARM executable sur le OS Linux

SDK (Software Development Kit) http://dl.altera.com/opencl/



AOCL-16.1.0.196-linux.tar
(20.3 GB)
aocl-rte-16.1.0-1.arm32.tgz
(1 MB)

Update: 16.1.1.200

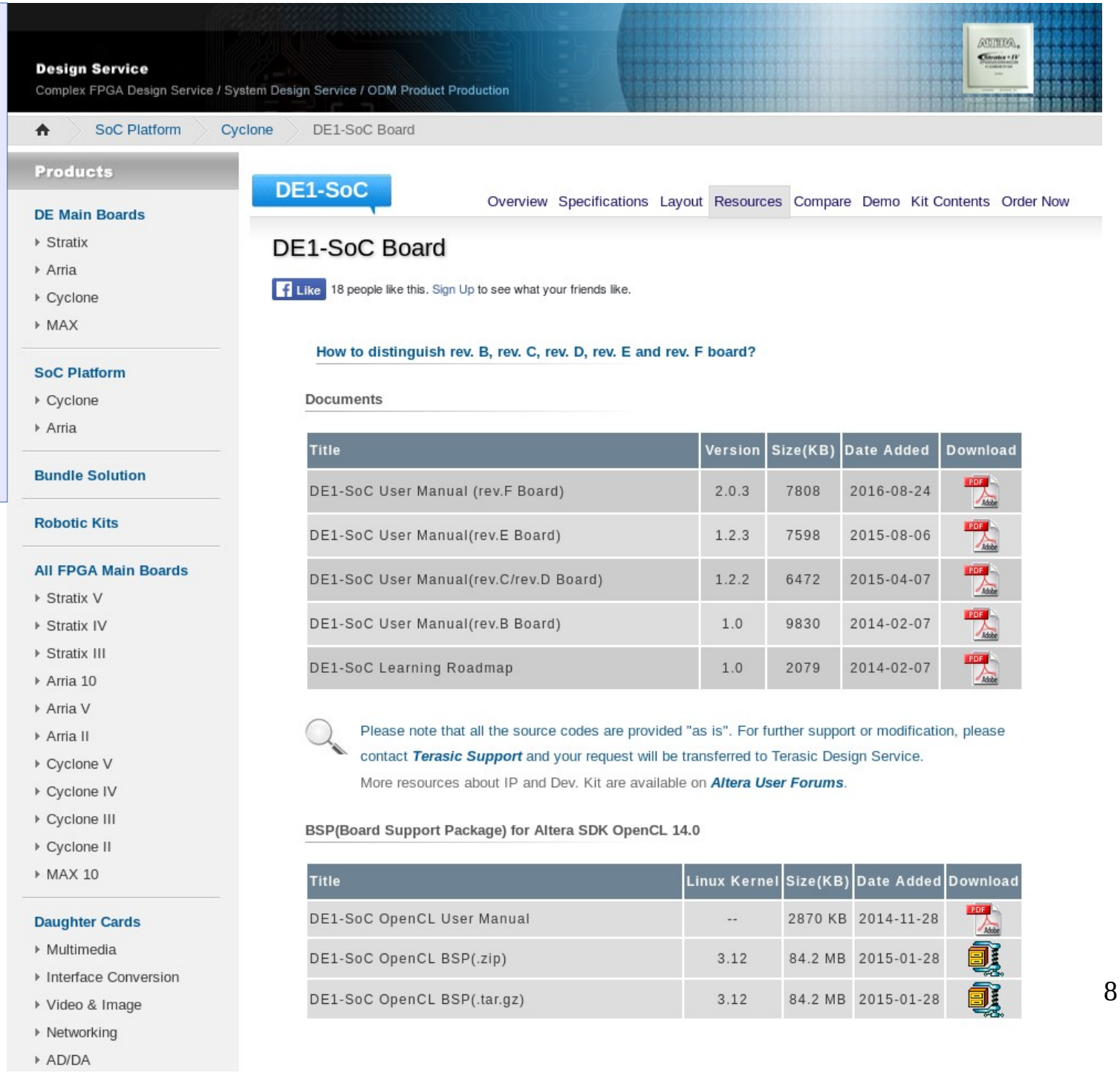
clralicepc11 (root)

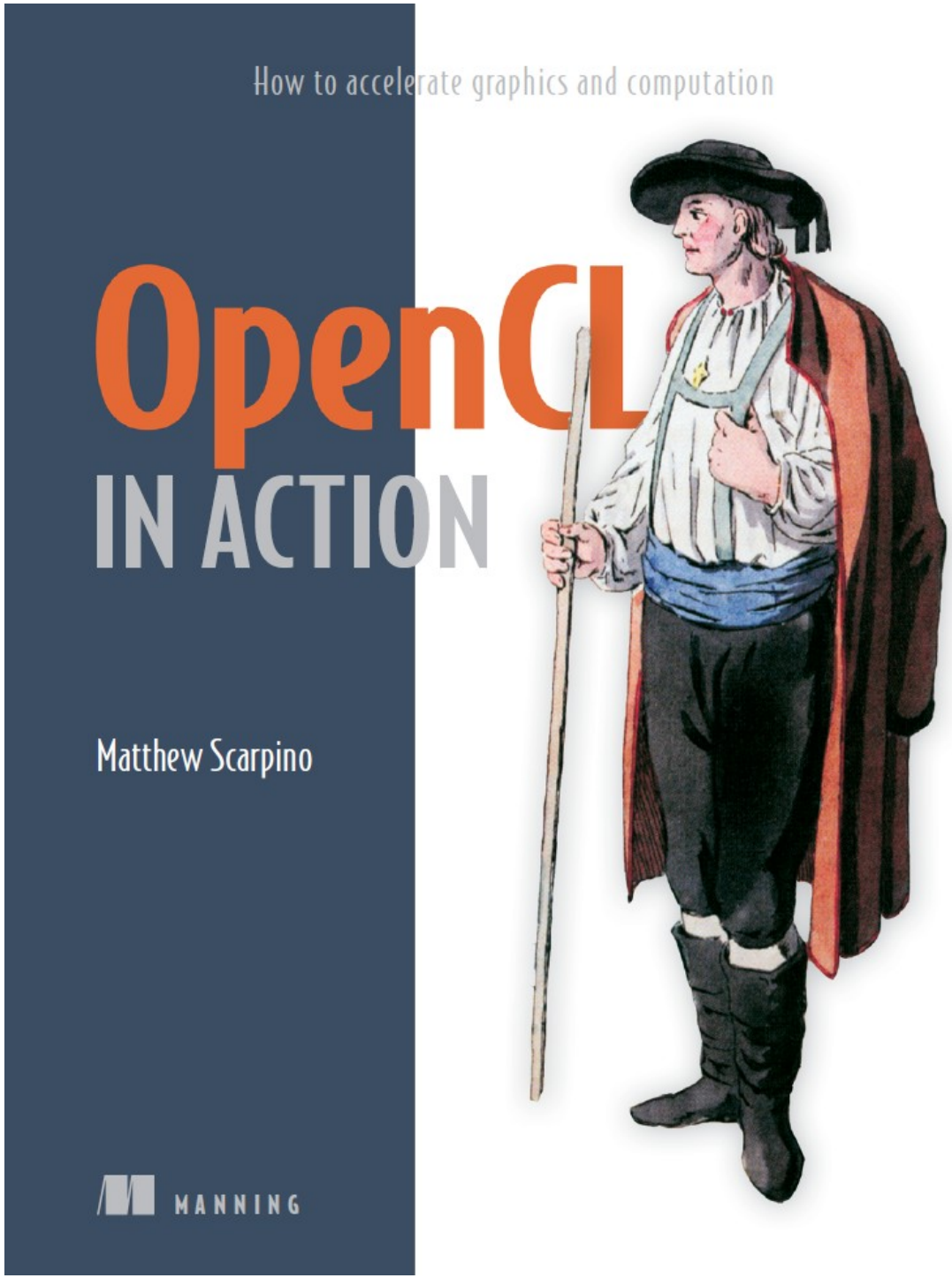
cd-de1-soc.terasic.com/

BSP
Board Support Package
pour SDK 16.0

DE1-SoC_openCL_BSP.zip
(200 MB)

contient une image Linux
pour la carte MicroSD





OpenCL (Open Computing Language) est la combinaison d'une API et d'un langage de programmation dérivé du C, proposé comme un standard ouvert par le Khronos Group. OpenCL est conçu pour programmer des systèmes parallèles hétérogènes comprenant par exemple à la fois un CPU multi-cœur et un GPU. OpenCL propose donc un modèle de programmation se situant à l'intersection naissante entre le monde des CPU et des GPU, les premiers étant de plus en plus parallèles, les seconds étant de plus en plus programmables.

(wikipedia FR)

CPU, GPU, FPGA, etc. = device

KHRONOS GROUP

The function

cl_program **clCreateProgramWithBinary** (cl_context *context*,
cl_uint *num_devices*,
const cl_device_id **device_list*,
const size_t **lengths*,
const unsigned char ***binaries*,
cl_int **binary_status*,
cl_int **errcode_ret*)

creates a program object for a context, and loads the binary bits specified by *binary* into the program object.

The **OpenCL** Specification

Version: 1.2

Document Revision: 19

Khronos OpenCL Working Group

Editor: Aaftab Munshi

Programme hôte (.c, .cc, .cpp, .cxx, .h)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#ifdef MAC
#include <OpenCL/cl.h>
#else
#include <CL/cl.h>
#endif

int main() {

   cl_platform_id platform;
   cl_device_id *devices;
   cl_uint num_devices, addr_data;
   cl_int i, err;
   char name_data[48], ext_data[4096];

   err = clGetPlatformIDs(1, &platform, NULL);
   if(err < 0) {
      perror("Couldn't find any platforms");
      exit(1);
   }

   err = clGetDeviceIDs(platform, CL_DEVICE_TYPE_ALL,
         1, NULL, &num_devices);
   if(err < 0) {
      perror("Couldn't find any devices");
      exit(1);
   }
```

Access first platform

Determine number of devices

Programme device
(kernel, .cl)

```
__kernel void reduction_vector(__global float* data,
      __local float* partial_sums, __global float* out) {

   int lid = get_local_id(0);
   int group_size = get_local_size(0);

   partial_sums[lid] = data[get_global_id(0)];
   barrier(CLK_LOCAL_MEM_FENCE);

   for(int i = group_size/2; i>0; i >>= 1) {
      if(lid < i) {
         partial_sums[lid] += partial_sums[lid + i];
      }
      barrier(CLK_LOCAL_MEM_FENCE);
   }

   if(lid == 0) {
      out[get_group_id(0)] = partial_sums[0];
   }

}

//_____________________________________________________________________________
__kernel void reduction_complete(__global float* data,
      __local float* partial_sums, __global float* sum) {

   int lid = get_local_id(0);
   int group_size = get_local_size(0);

   partial_sums[lid] = data[get_local_id(0)];
   barrier(CLK_LOCAL_MEM_FENCE);

   for(int i = group_size/2; i>0; i >>= 1) {
      if(lid < i) {
         partial_sums[lid] += partial_sums[lid + i];
      }
      barrier(CLK_LOCAL_MEM_FENCE);
   }

   if(lid == 0) {
      *sum = partial_sums[0];
   }

}
```

Analogie avec une école

école	*device* OpenCL (CPU, GPU, FPGA, ...)
problème à résoudre	programme *kernel*
salle de classe	composante de calcul (coeur CPU, ...)
classe d'élèves	group de travail (*work-group*)
élève dans une classe	unité de calcul (*work-item*)

mémoire = globale (table noire centrale), locale (table noire dans la classe), privée (notebook)

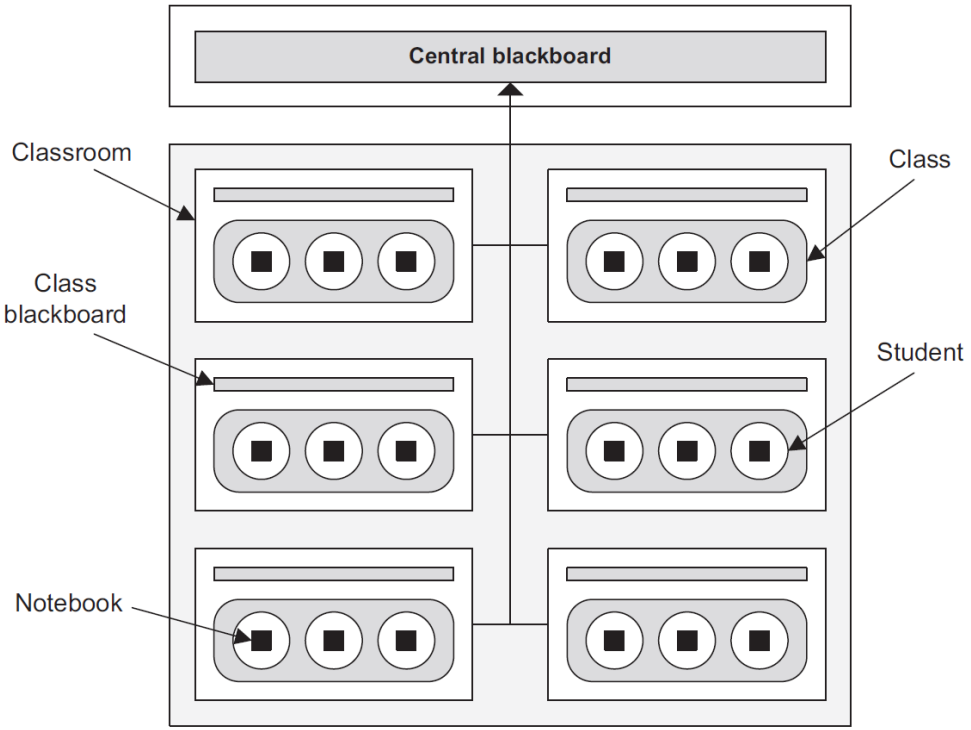


Figure 4.6 **School of math students in OpenCL device analogy**

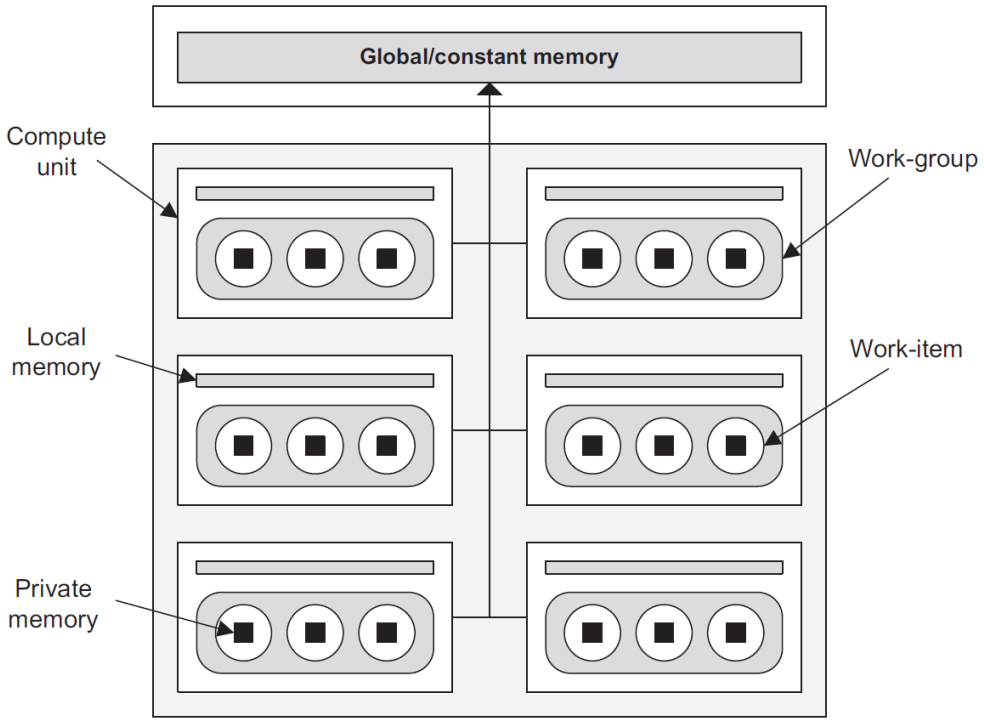


Figure 4.7 **OpenCL device model**

Exemple de réduction numérique

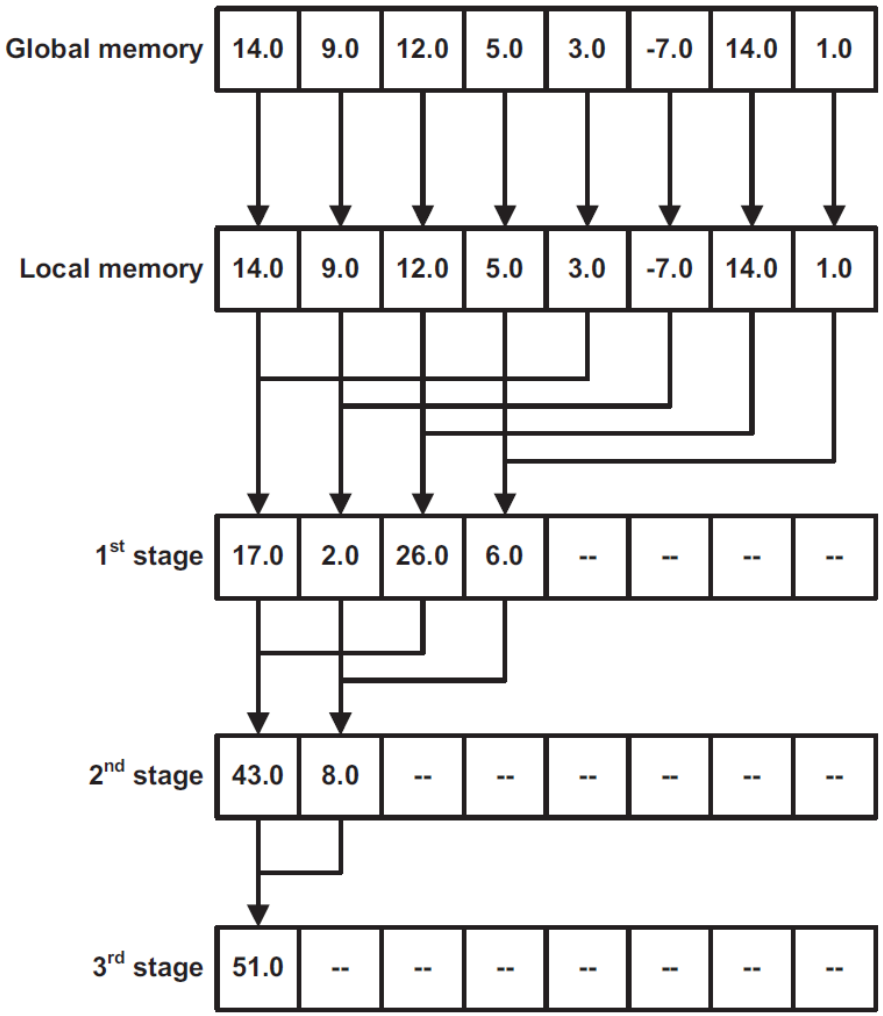


Figure 10.1
Multistage reduction in OpenCL

```
sum = 0.0;
for (int i=0; i<1048576; i++)
{
  sum += data[i];
}
```

```
__kernel void reduction_scalar (
     __global float* data,
     __local  float* partial_sums,
     __global float* output)
{...}
```

```
__kernel void reduction_vector (
     __global float4* data,
     __local  float4* partial_sums,
     __global float4* output)
{...}
```

Note: 1048576 (2^{20}) *work-items* dans 4096 *work-groups* (256 *work-items* par *work-group*)

OpenCL sur FPGA (DE1-SoC), un exemple

1) PC avec le SDK OpenCL (Quartus, etc.)

Compilation kernel (~35 minutes):

```
> aoc reduction.cl -o reduction.aocx --board de1soc_sharedonly
```

(`.aocx` = image FPGA)

Compilation (croisée) du code hôte:

```
> arm-linux-gnueabihf-g++ reduction.cxx -o reduction \
-I/users_local1/root/intelFPGA/16.1/hld/host/include \
-L/users_local1/root/intelFPGA/16.1/hld/board/de1soc/arm32/lib \
-L/users_local1/root/intelFPGA/16.1/hld/host/arm32/lib \
-lalteracl -lalterammdpcie
```

2) DE1-SoC board (RTE même version que le SDK)

```
> aocl diagnose
> aocl program /dev/acl0 reduction.aocx
> ./reduction
```

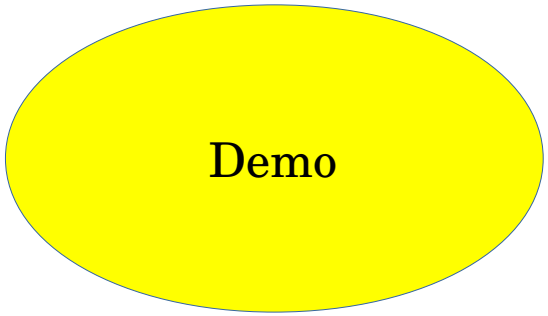
Rapport à la fin de la compilation

HLD FPGA Reports (Beta) View reports...

Area report (system view)
(area utilization values are estimated)
Notation *file:X* > *file:Y* indicates a function call on line X was inlined using code on line Y.

	ALUTs	FFs	RAMs	DSPs	Details
Kernel System (Logic: 30%)	**20591 (19%)**	**28505 (13%)**	**99 (19%)**	**0 (0%)**	
Board interface	2160	1908	20	0	• Platform i...
Global interconnect	9588	10682	0	0	• Global int...
reduction_complete	**4287 (4%)**	**7403 (3%)**	**40 (8%)**	**0 (0%)**	• **Number of ...**
Function overhead	1618	1777	0	0	• Kernel dis...
Block4.wii_blk	64 (0%)	64 (0%)	0 (0%)	0 (0%)	
Block5	656 (1%)	965 (0%)	14 (3%)	0 (0%)	
State	163	570	1	0	• Resources ...
Computation	493	395	13	0	
reduction_complete_scal_1.cl:31	383	322	13	0	
Load	301	298	13	0	
Pointer Computation (x2)	48	0	0	0	
Store	34	24	0	0	
reduction_complete_scal_1.cl:32	110	73	0	0	
Block6	1285 (1%)	2178 (1%)	8 (2%)	0 (0%)	
Block7	664 (1%)	2419 (1%)	18 (4%)	0 (0%)	
reduction_vector	**4556 (4%)**	**8512 (4%)**	**39 (8%)**	**0 (0%)**	• **Number of ...**
Function overhead	1618	1777	0	0	• Kernel dis...
Block0.wii_blk	128 (0%)	128 (0%)	0 (0%)	0 (0%)	

(et 11% pour “hello_world”)

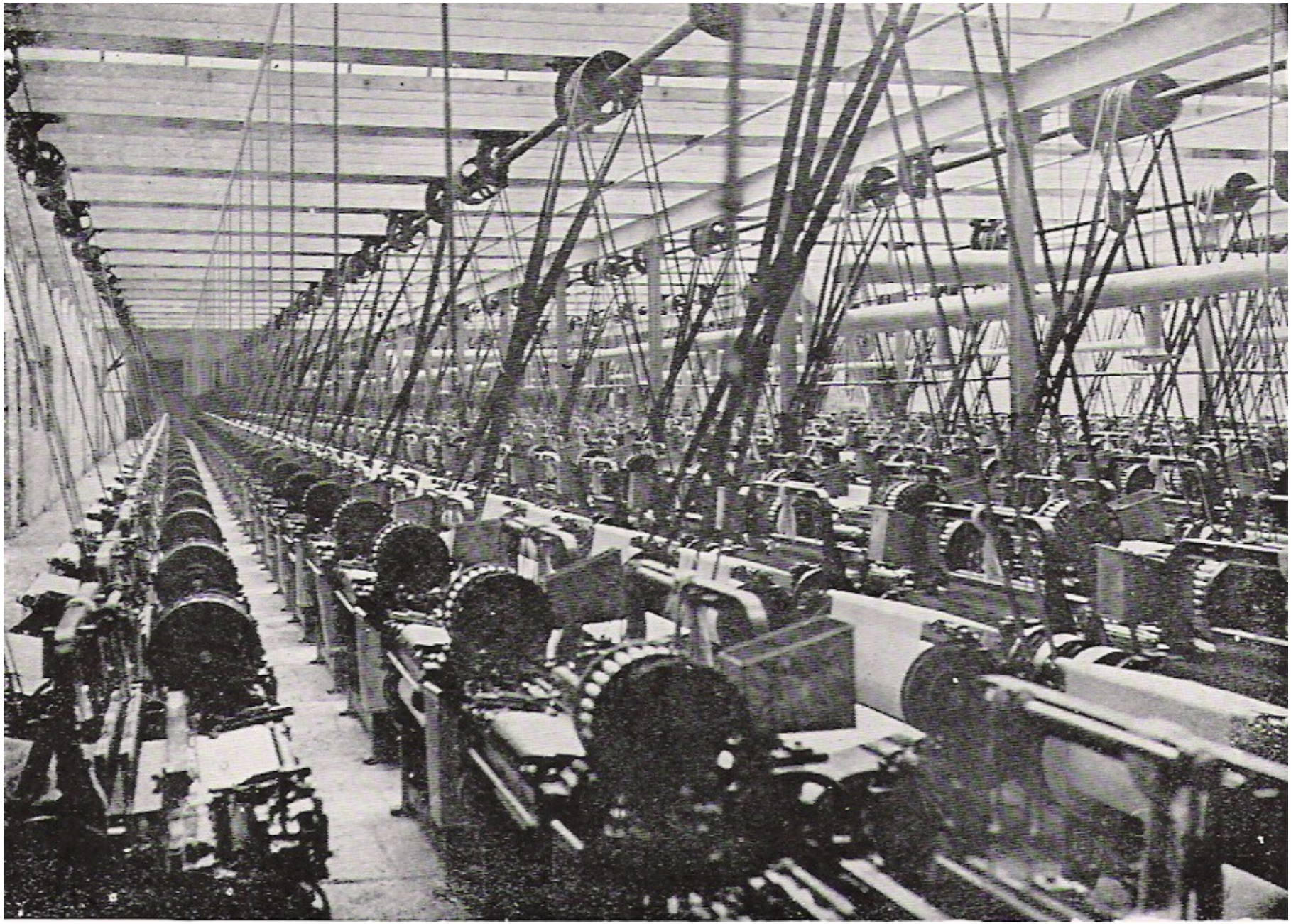

Demo

Quel parallélisme pour le FPGA?

- l'exécution parallèle est *innée* à la construction
- granularité au niveau des cellules logiques qui composent les opérations
- parallèlisme par chaîne de traitement (*pipeline*)
- nombre arbitraire de régistres mémoire
- création et utilisations des bibliothèques OpenCL (RTL)
- optimisations différentes par rapport aux CPUs et GPUs

Biblio:
[1] DE1-SoC User Manual
[2] Intel FPGA SDK for OpenCL Cyclone V SoC Getting Started Guide
[3] Intel FPGA SDK for OpenCL Getting Started Guide
[4] Intel FPGA SDK for OpenCL Programming Guide
[5] Intel FPGA SDK for OpenCL Best Practices Guide
[6] Intel FPGA RTE for OpenCL, Getting Started Guide
[7] Altera Using Linux on the DE1-SoC
[8] Khronos The OpenCL Specification 1.1, 1.2
[9] OpenCL in Action

Merci de votre attention



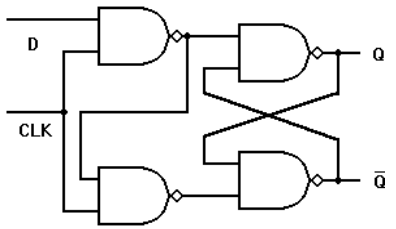
FPGA ?

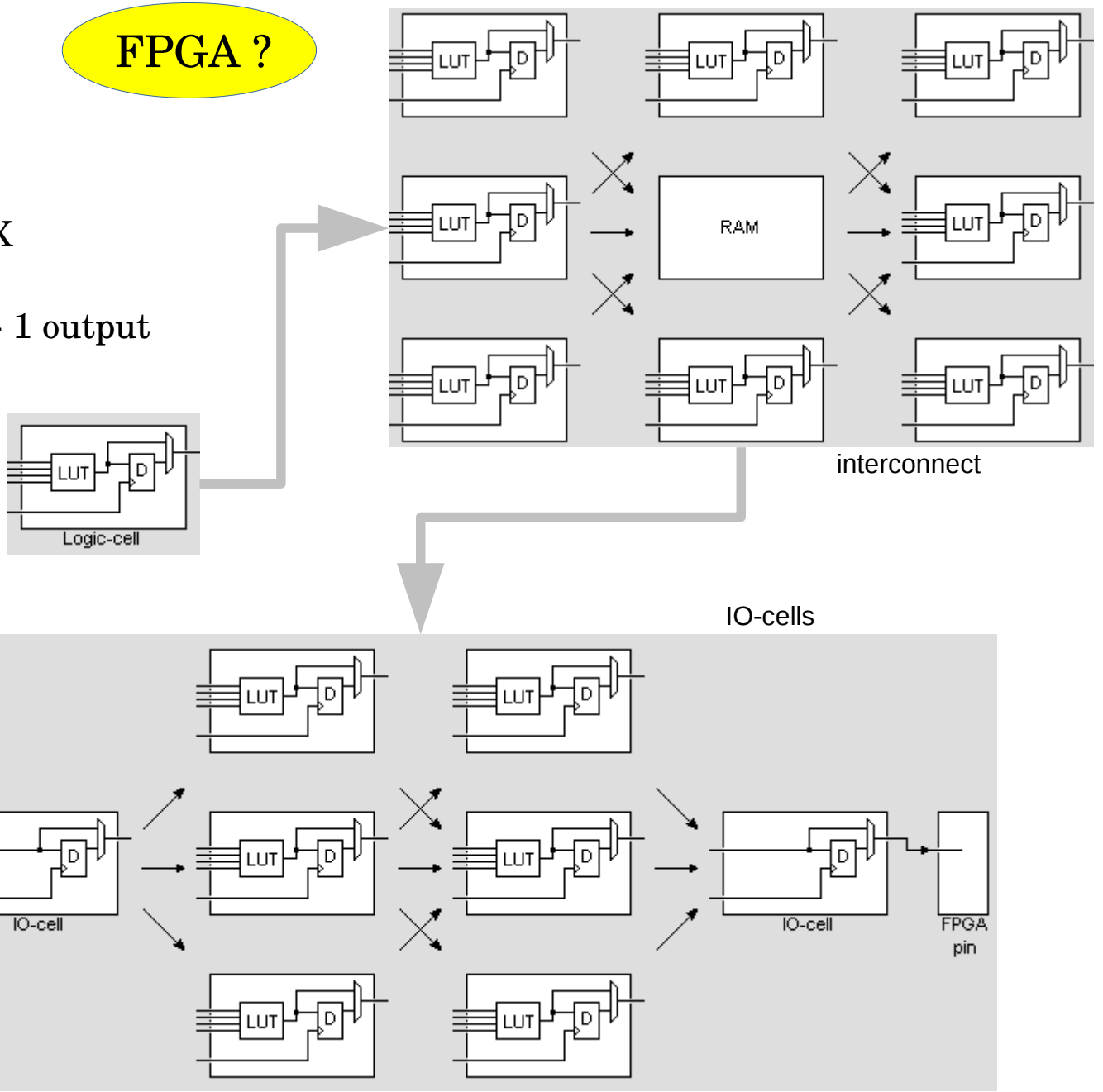
- FPGA = N[large] x Logic Cells
- Logic Cell = LUT + Register + MUX
- Look-Up Tables (LUT) = n inputs + 1 output

Ex. LUT doing an AND gate:

In[1:0]	Out
00	0
01	0
10	0
11	1

- D flip-flop (memory cell)





+ multiple clocks (domains).

http://www.fpga4fun.com

DE1-SoC Cyclone V

FPGA:

- Dual-core ARM Cortex-A9 (HPS)
- 85K Programmable Logic Elements
- 4,450 Kbits embedded memory

Memory:

- 64MB SDRAM on FPGA
- 1GB DDR3 SDRAM on HPS
- Micro SD Card Socket on HPS

Communication:

- USB 2.0
- 10/100/1000 Ethernet
- PS/2 mouse/keyboard

Display:

- 24-bit VGA

Indicators:

- Six 7-segment displays

Power:

- 12V DC

OpenCL:

- Altera SDK OpenCL 14.0
- Linux Micro SD Card Image
- User Manual

Connectors:

- 40-pin FPGA expansion