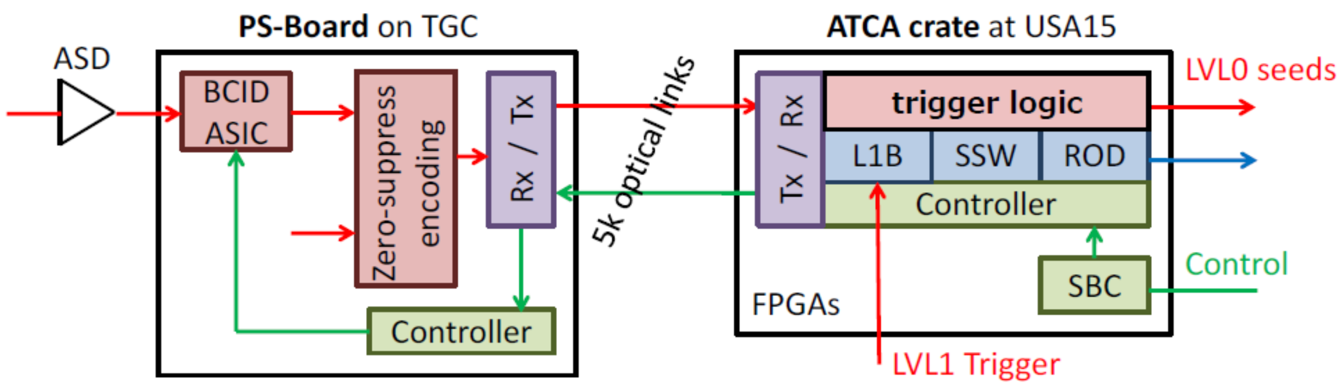
Zynqと組み込みLinuxによるフロントエンドエレクトロニクス制御

東大素セ, 高エ研[A], Open-It
坂本宏, 佐々木修[A], 池野正弘[A]

目次

- フロントエンドエレクトロニクスへの要請
- ディジタルエレクトロニクス技術動向
- SoC(System on Chip)デバイスの紹介
- ソフトウエア-ハードウエア-コデザイン
- Zynq7000シリーズの技術評価
- 組み込みシステムとしてのパフォーマンス
- ベアメタルソフトウエアによる評価
- Linuxアプリケーションによる評価
- 今後の開発課題

TGC L1 trigger electronics scheme for Phase- II



✓ ASIC's for PS-Board

➡ LVDS, variable delay, BCID, test pulse generator

➡ Zero-suppress and encoding logic of hit signals and interface to GB transfer

✓ module with FPGAs for trigger/readout at USA15

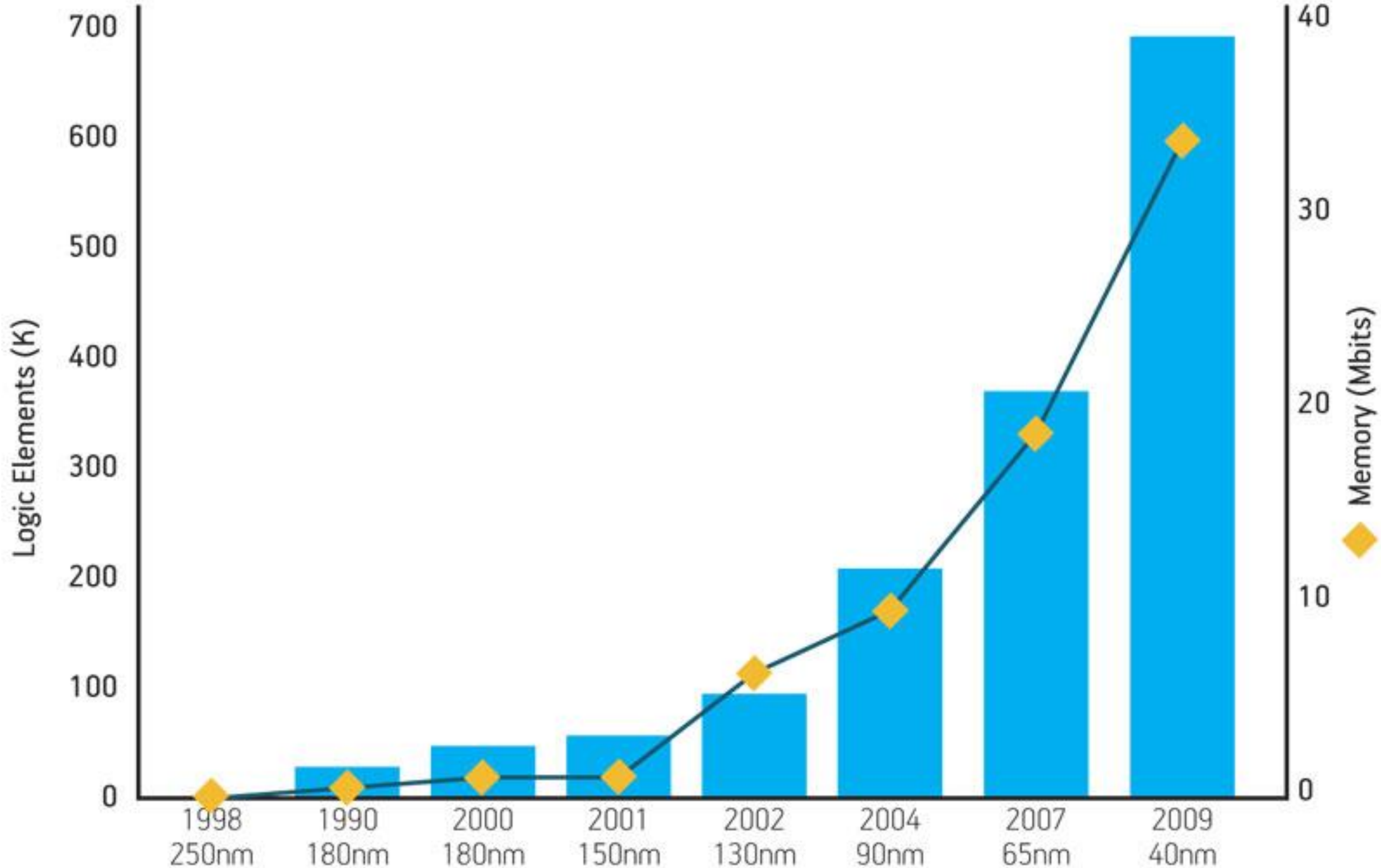
➡ output LVL0 seeds for MDT level-0 trigger logic

➡ receive LVL1 trigger signals for readout

  ➡ long L1-buffer memory to cope with L1 latency(60 μs)

➡ **optimize the trigger logic**

14 /12



The rapid increase in FPGA complexity

http://www.techdesignforums.com/practice/technique/device-native-debug-and-verification-for-fpga/

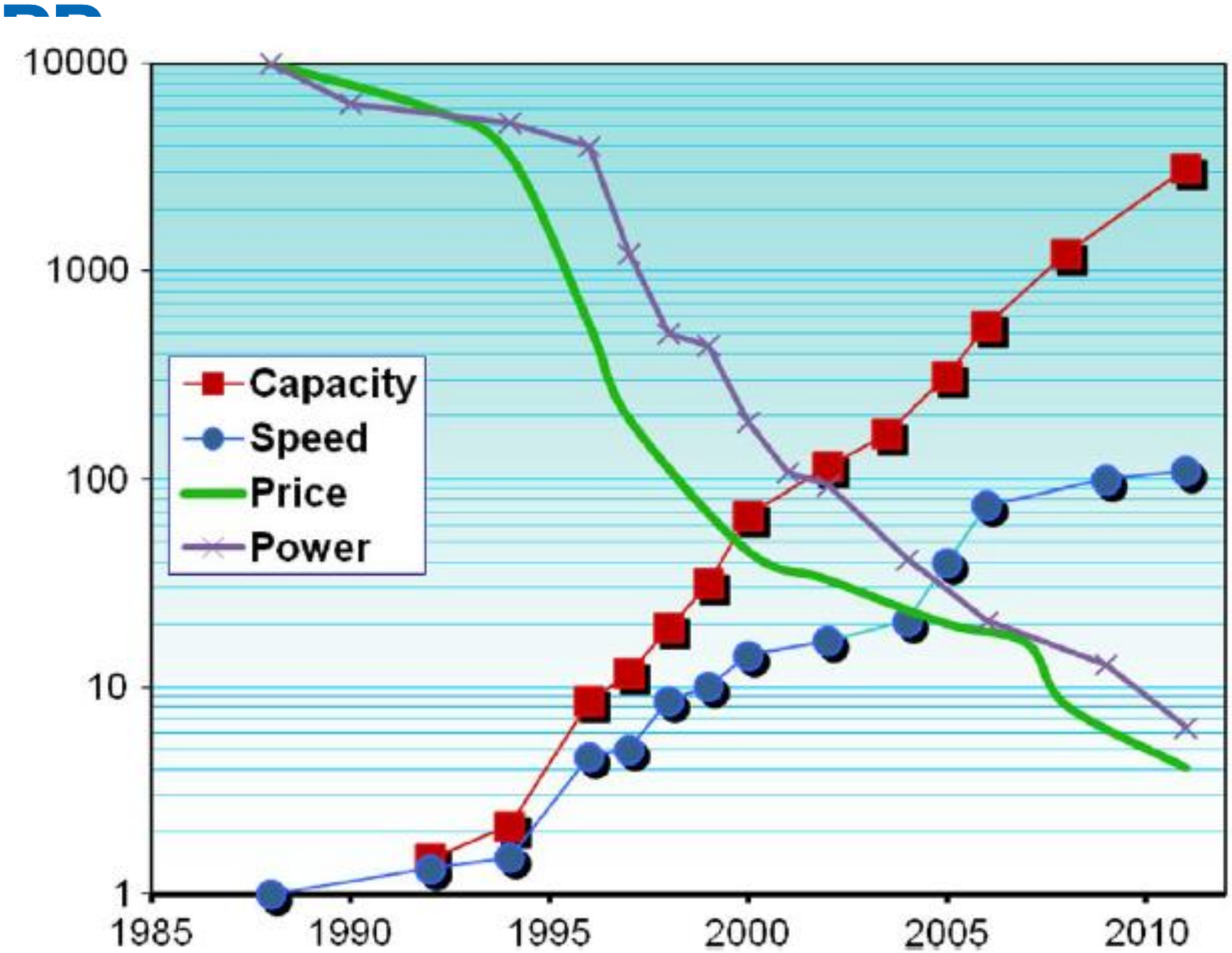


Fig. 1. ***Xilinx FPGA attributes relative to 1988. Capacity is logic cell count. Speed is same-function performance in programmable fabric. Price is per logic cell. Power is per logic cell. Price and power are scaled up by 10 000×. Data: Xilinx published data.***

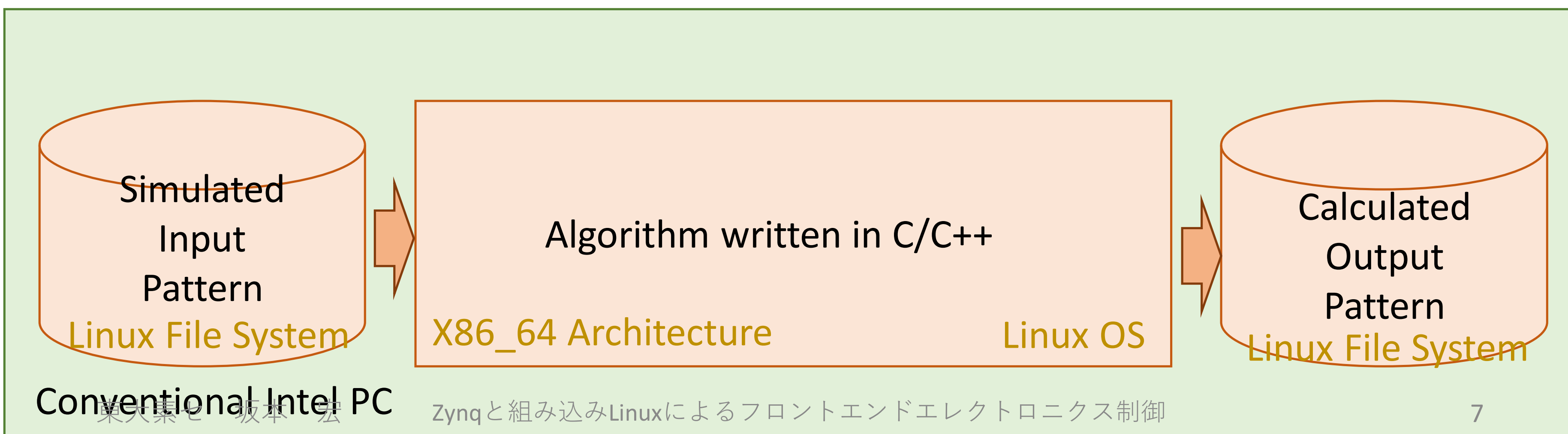
Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology
Stephen M. (Steve) Trimberger, Proceedings of the IEEE | Vol. 103, No. 3, March 2015

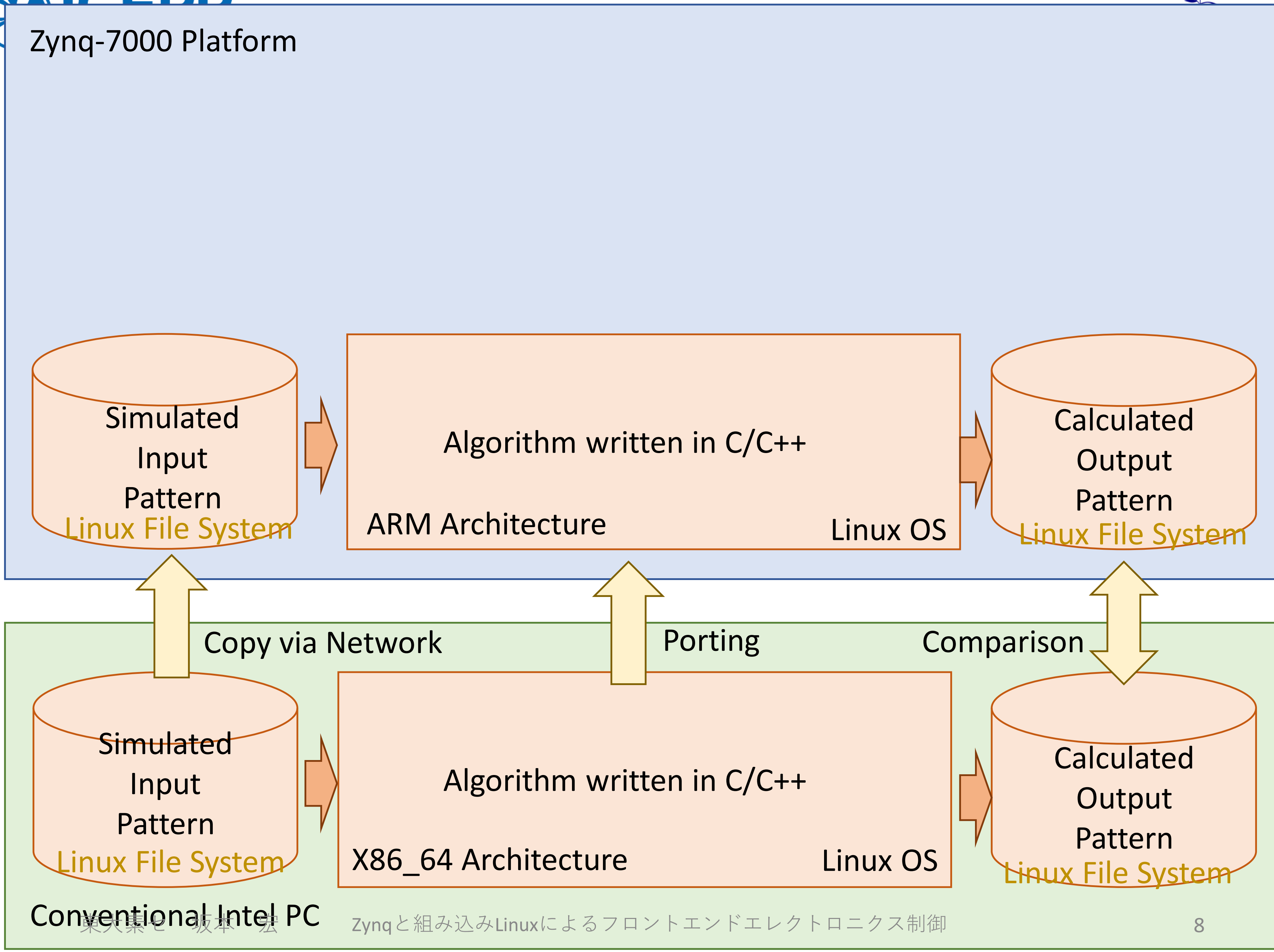
System on Chip (SoC) Device

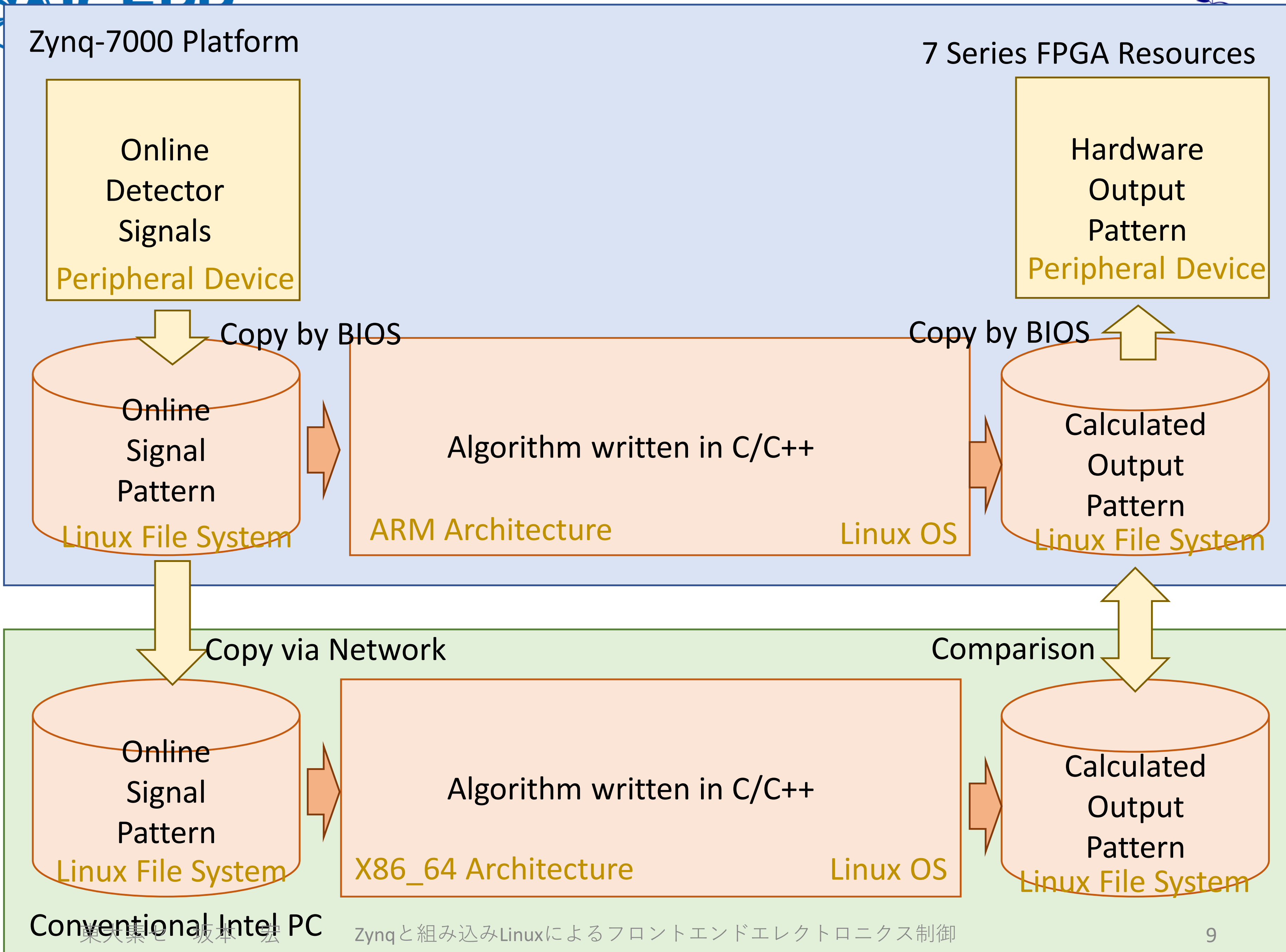
- 大規模化するFPGA
  - デザインの開発にかかるコスト
  - 複雑大規模なデザインのデバッグの困難
  - 診断や制御機構の組み込みのオーバーヘッド
- これらの問題を解決するSoCデバイス
  - 通常のProgrammable Logic (PL)
  - 標準的なCPUコア(ARMなど)によるProcessing System (PS)
  - 規格化された周辺装置インターフェースとそのIntellectual Property (IP)
  - 統合された開発環境

Software-Hardware Co-design

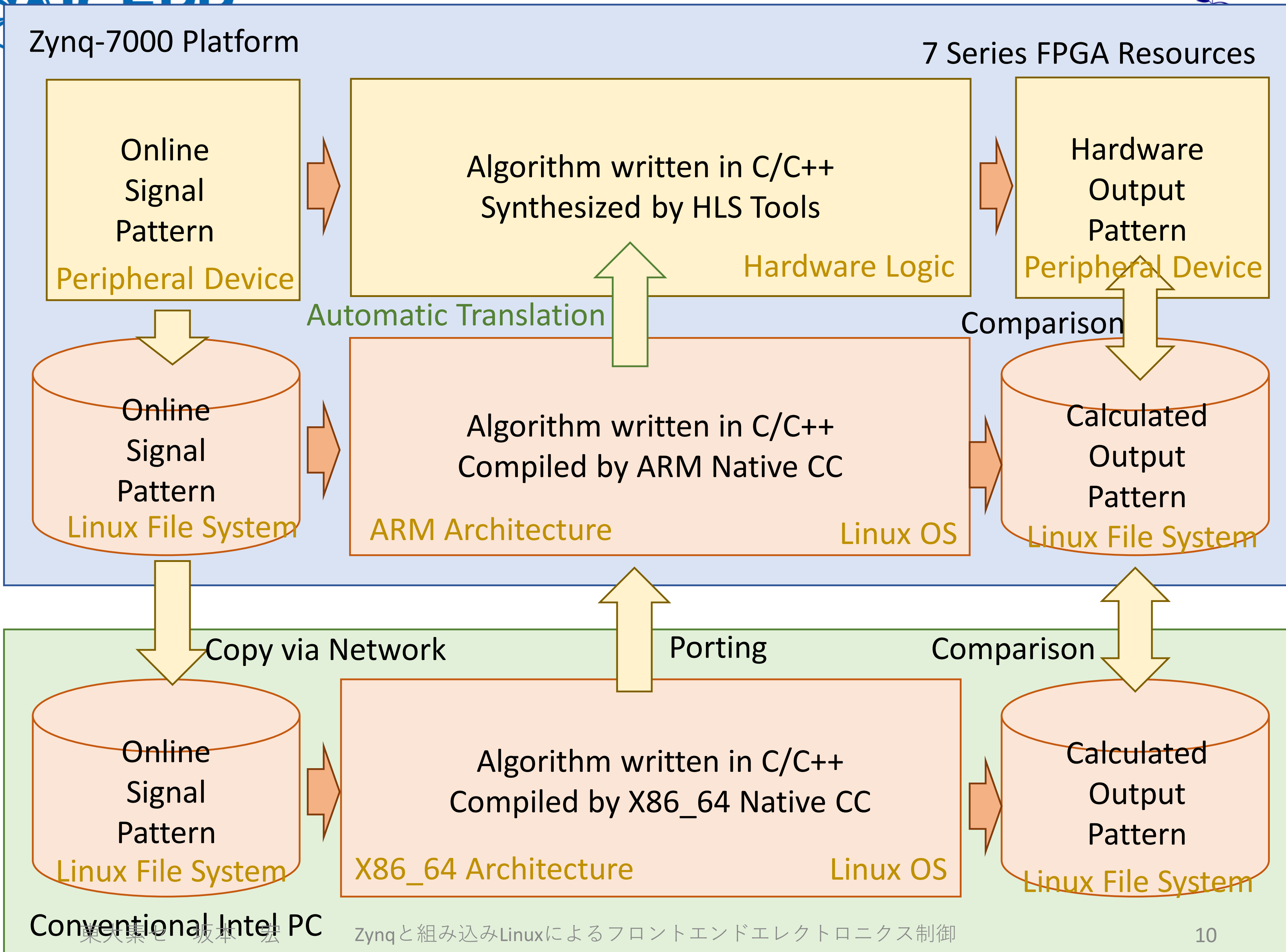
- Start algorithm design using C/C++ on conventional PC
- Port the C/C++ algorithm to Zynq
- Convert the algorithm into HDL (Manually/Automatic)
- Validate the hardware implementation



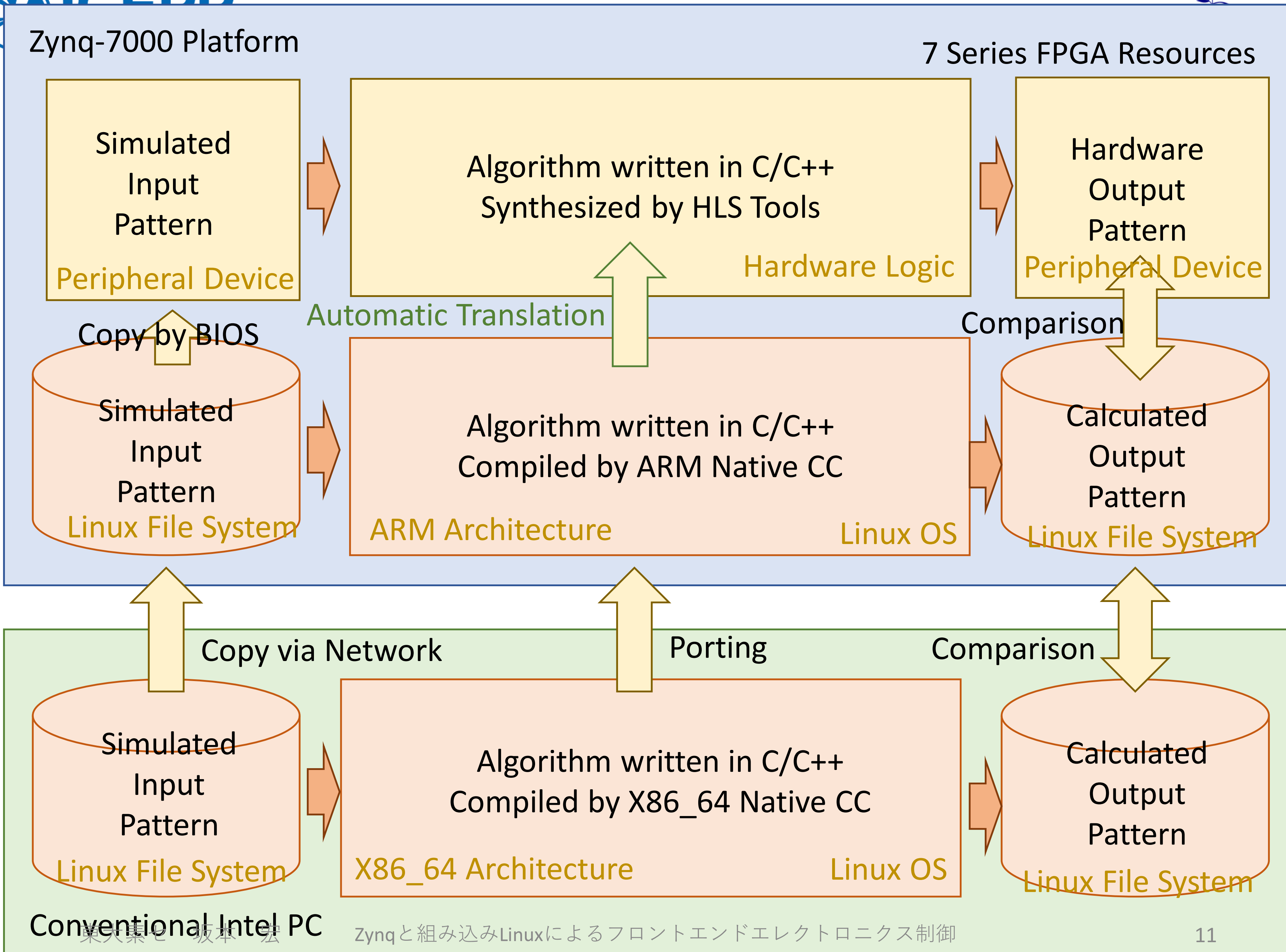

Zynq-7000 Platform
Simulated
Input
Pattern
Linux File System
Algorithm written in C/C++
ARM Architecture
Linux OS
Calculated
Output
Pattern
Linux File System
Copy via Network
Porting
Comparison
Simulated
Input
Pattern
Linux File System
Algorithm written in C/C++
X86_64 Architecture
Linux OS
Calculated
Output
Pattern
Linux File System
Conventional Intel PC



Zynq-7000 Platform
7 Series FPGA Resources
Online Detector Signals
Peripheral Device
Hardware Output Pattern
Peripheral Device
Copy by BIOS
Copy by BIOS
Online Signal Pattern
Linux File System
Algorithm written in C/C++
ARM Architecture
Linux OS
Calculated Output Pattern
Linux File System
Copy via Network
Comparison
Online Signal Pattern
Linux File System
Algorithm written in C/C++
X86_64 Architecture
Linux OS
Calculated Output Pattern
Linux File System
Conventional Intel PC



Zynq-7000 Platform
7 Series FPGA Resources
Online Signal Pattern
Peripheral Device
Algorithm written in C/C++
Synthesized by HLS Tools
Hardware Logic
Hardware Output Pattern
Peripheral Device
Automatic Translation
Comparison
Online Signal Pattern
Linux File System
Algorithm written in C/C++
Compiled by ARM Native CC
ARM Architecture
Linux OS
Calculated Output Pattern
Linux File System
Copy via Network
Porting
Comparison
Online Signal Pattern
Linux File System
Algorithm written in C/C++
Compiled by X86_64 Native CC
X86_64 Architecture
Linux OS
Calculated Output Pattern
Linux File System
Conventional Intel PC


Zynq-7000 Platform
7 Series FPGA Resources
Simulated Input Pattern
Peripheral Device
Algorithm written in C/C++
Synthesized by HLS Tools
Hardware Logic
Hardware Output Pattern
Peripheral Device
Copy by BIOS
Automatic Translation
Comparison
Simulated Input Pattern
Linux File System
Algorithm written in C/C++
Compiled by ARM Native CC
ARM Architecture
Linux OS
Calculated Output Pattern
Linux File System
Copy via Network
Porting
Comparison
Simulated Input Pattern
Linux File System
Algorithm written in C/C++
Compiled by X86_64 Native CC
X86_64 Architecture
Linux OS
Calculated Output Pattern
Linux File System
Conventional Intel PC

Zynq-7000 All Programmable SoC Overview, DS190 (v1.8) May 27, 2015, Xilinx Inc.

Table 1: **Zynq-7000 All Programmable SoC *(Cont'd)*** Zed Board ZC706

	Zynq-7000 All Programmable SoC							
	Device Name	Z-7010	Z-7015	Z-7020	Z-7030	Z-7035	Z-7045	Z-7100
	Part Number	XC7Z010	XC7Z015	XC7Z020	XC7Z030	XC7Z035	XC7Z045	XC7Z100
Programmable Logic	Xilinx 7 Series Programmable Logic Equivalent	Artix®-7 FPGA	Artix-7 FPGA	Artix-7 FPGA	Kintex®-7 FPGA	Kintex-7 FPGA	Kintex-7 FPGA	Kintex-7 FPGA
	Programmable Logic Cells (Approximate ASIC Gates)(3)	28K Logic Cells (~430K)	74K Logic Cells (~1.1M)	85K Logic Cells (~1.3M)	125K Logic Cells (~1.9M)	275K Logic Cells (~4.1M)	350K Logic Cells (~5.2M)	444K Logic Cells (~6.6M)
	Look-Up Tables (LUTs)	17,600	46,200	53,200	78,600	171,900	218,600	277,400
	Flip-Flops	35,200	92,400	106,400	157,200	343,800	437,200	554,800
	Extensible Block RAM (# 36 Kb Blocks)	240 KB (60)	380 KB (95)	560 KB (140)	1,060 KB (265)	2,000 KB (500)	2,180 KB (545)	3,020 KB (755)
	Programmable DSP Slices (18x25 MACCs)	80	160	220	400	900	900	2,020
	Peak DSP Performance (Symmetric FIR)	100 GMACs	200 GMACs	276 GMACs	593 GMACs	1,334 GMACs	1,334 GMACs	2,622 GMACs
	PCI Express® (Root Complex or Endpoint)(4)	—	Gen2 x4	—	Gen2 x4	Gen2 x8	Gen2 x8	Gen2 x8

Kintex-7 FPGA Feature Summary

Table 5: **Kintex-7 FPGA Feature Summary by Device**

Device	Logic Cells	Configurable Logic Blocks (CLBs)		DSP Slices(2)	Block RAM Blocks(3)			CMTs(4)	PCIe(5)	GTXs	XADC Blocks	Total I/O Banks(6)	Max User I/O(7)
		Slices(1)	Max Distributed RAM (Kb)		18 Kb	36 Kb	Max (Kb)						
XC7K70T	65,600	10,250	838	240	270	135	4,860	6	1	8	1	6	300
XC7K160T	162,240	25,350	2,188	600	650	325	11,700	8	1	8	1	8	400
XC7K325T	326,080	50,950	4,000	840	890	445	16,020	10	1	16	1	10	500
XC7K355T	356,160	55,650	5,088	1,440	1,430	715	25,740	6	1	24	1	6	300
XC7K410T	406,720	63,550	5,663	1,540	1,590	795	28,620	10	1	16	1	10	500
XC7K420T	416,960	65,150	5,938	1,680	1,670	835	30,060	8	1	32	1	8	400
XC7K480T	477,760	74,650	6,788	1,920	1,910	955	34,380	8	1	32	1	8	400

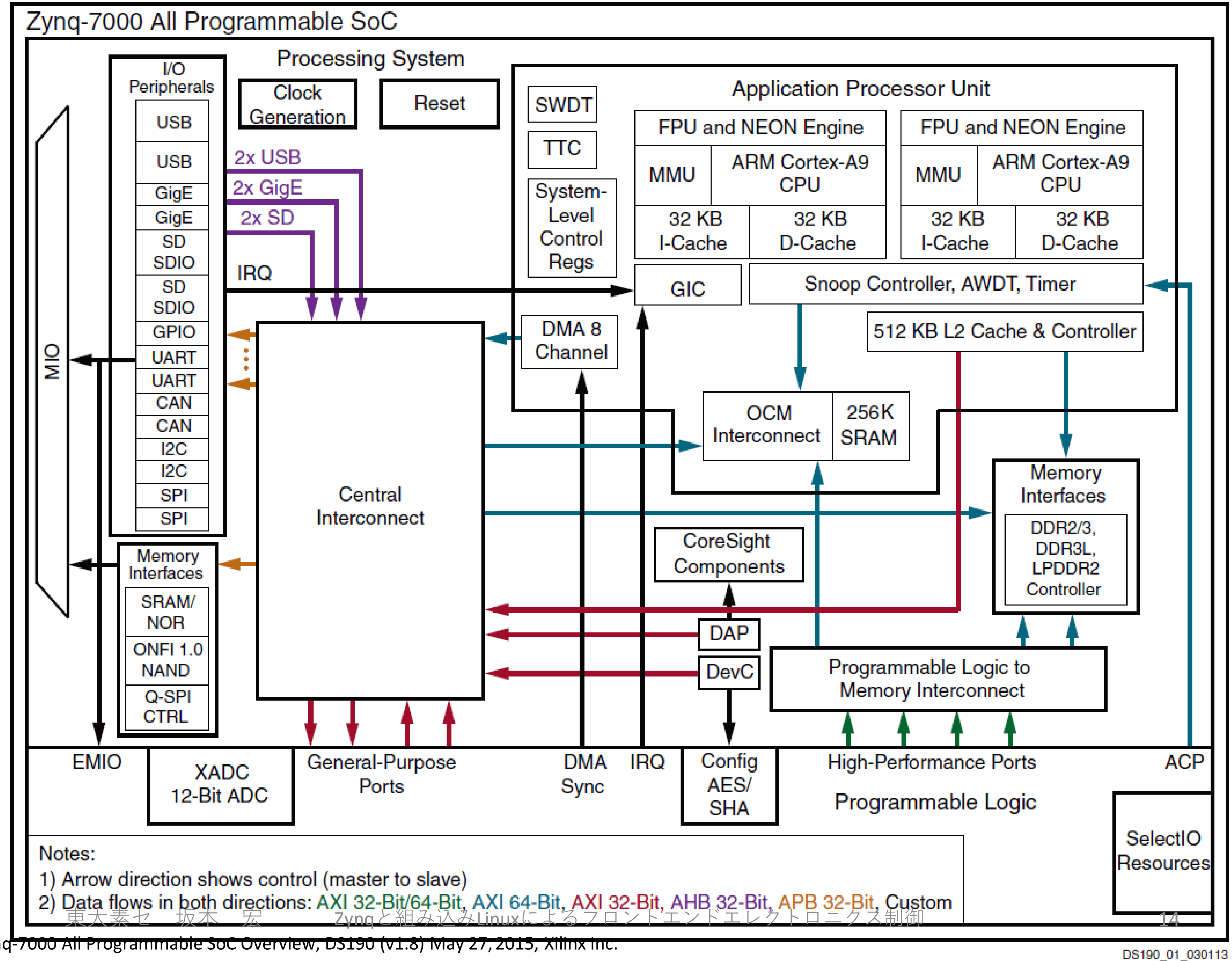
Table 2: Device-Package Combinations: Maximum I/Os and GTP and GTX Transceivers

Package[1]	CLG225			CLG400			CLG484			CLG485[2]				SBG485[2] SBV485			
Size	13 x 13 mm			17 x 17 mm			19 x 19 mm			19 x 19 mm				19 x 19 mm			
Ball Pitch	0.8 mm			0.8 mm			0.8 mm			0.8 mm				0.8 mm			
Transceiver Speed (max)										6.25 Gb/s				6.6 Gb/s			
Device	PS I/O	SelectIO HR[3]	SelectIO HP[4]	PS I/O	SelectIO HR[3]	SelectIO HP[4]	PS I/O	SelectIO HR[3]	SelectIO HP[4]	PS I/O	GTP	SelectIO HR[3]	SelectIO HP[4]	PS I/O	GTX	SelectIO HR[3]	SelectIO HP[4]
XC7Z010	86	54	–	130	100	–											
XC7Z015										130	4	150	–				
XC7Z020				130	125	–	130	200	–								
XC7Z030								Zed Board						130	4	50	100
XC7Z035																	
XC7Z045																	
XC7Z100																	

Package[1]	FBG484 FBV484				FBG676 FBV676				FFG676 FFV676				FFG900 FFV900				FFG1156 FFV1156			
Size	23 x 23 mm				27 x 27 mm				27 x 27 mm				31 x 31 mm				35 x 35 mm			
Ball Pitch	1.0 mm				1.0 mm				1.0 mm				1.0 mm				1.0 mm			
Transceiver Speed (max)	6.6 Gb/s				6.6 Gb/s				12.5 Gb/s				12.5 Gb/s				10.3 Gb/s			
Device	PS I/O	GTX	SelectIO HR[2]	SelectIO HP[3]	PS I/O	GTX	SelectIO HR[2]	SelectIO HP[3]	PS I/O	GTX	SelectIO HR[2]	SelectIO HP[3]	PS I/O	GTX	SelectIO HR[2]	SelectIO HP[3]	PS I/O	GTX	SelectIO HR[2]	SelectIO HP[3]
XC7Z010																				
XC7Z015																				
XC7Z020																				
XC7Z030	130	4	100	63	130	4	100	150	130	4	100	150								
XC7Z035					130	8	100	150	130	8	100	150	130	16	212	150				
XC7Z045					130	8	100	150	130	8	100	150	130	16	212	150				
XC7Z100													130	16 ZC706	212	150	130	16	250	150

Notes:
1. All packages listed are Pb-free (FBG and FFG with exemption 15). Some packages are available with a Pb option.
2. HR = High Range I/O with support for I/O voltage from 1.2V to 3.3V.
3. HP = High Performance I/O with support for I/O voltage from 1.2V to 1.8V.

Zynq-7000 All Programmable SoC Overview, DS190 (v1.8) May 27, 2015, Xilinx Inc.



Zynq-7000 All Programmable SoC Overview, DS190 (v1.8) May 27, 2015, Xilinx Inc.

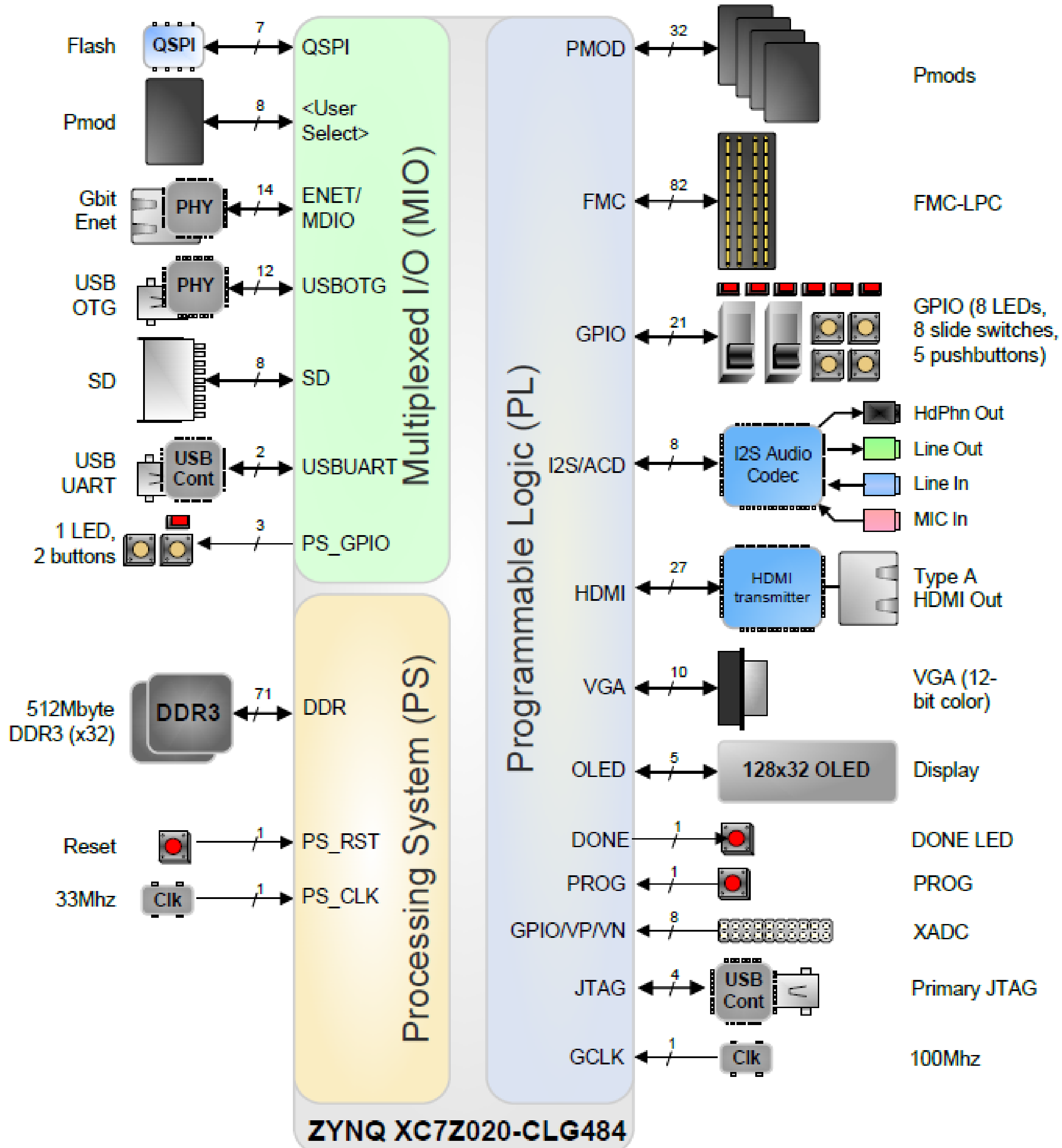
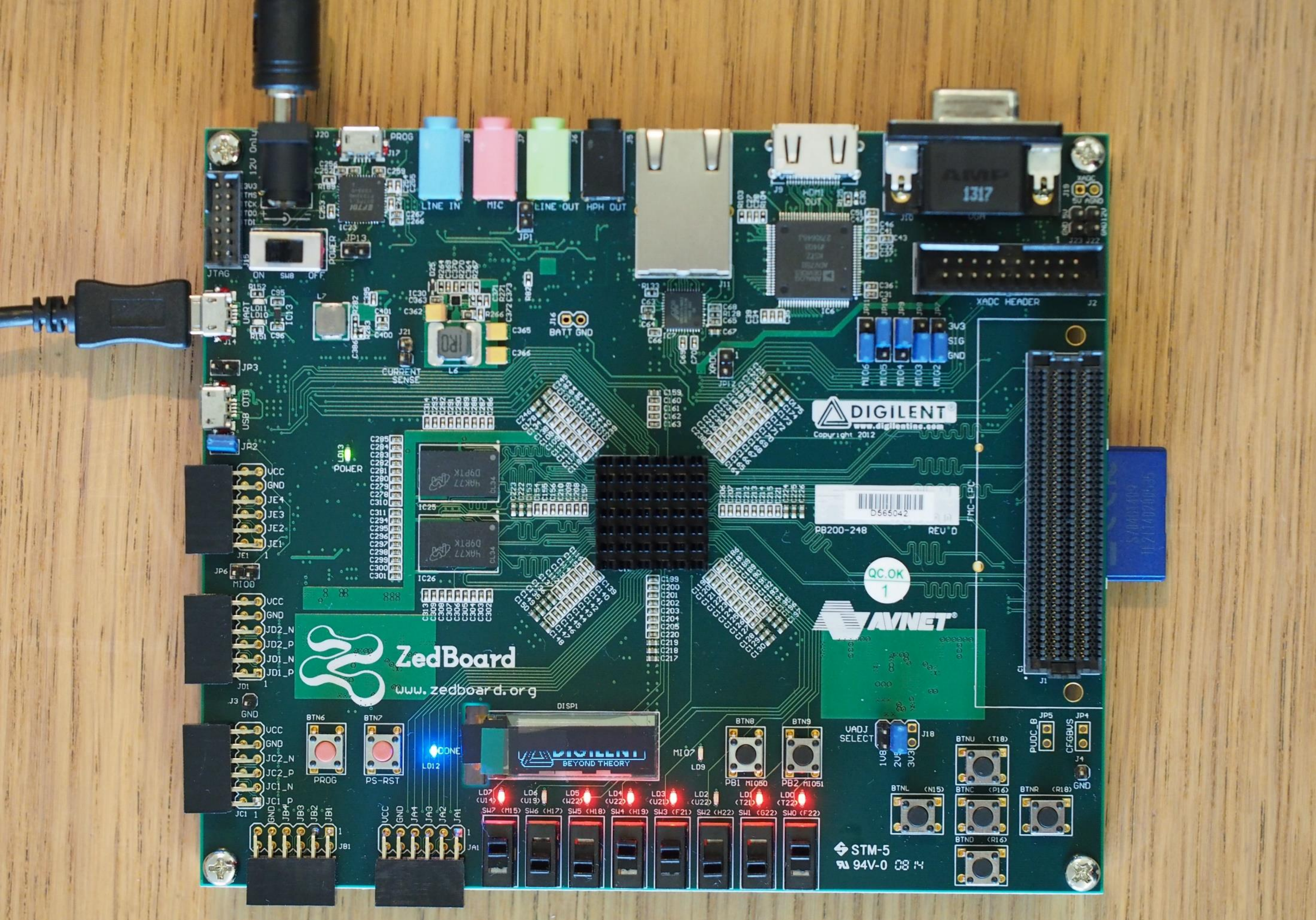
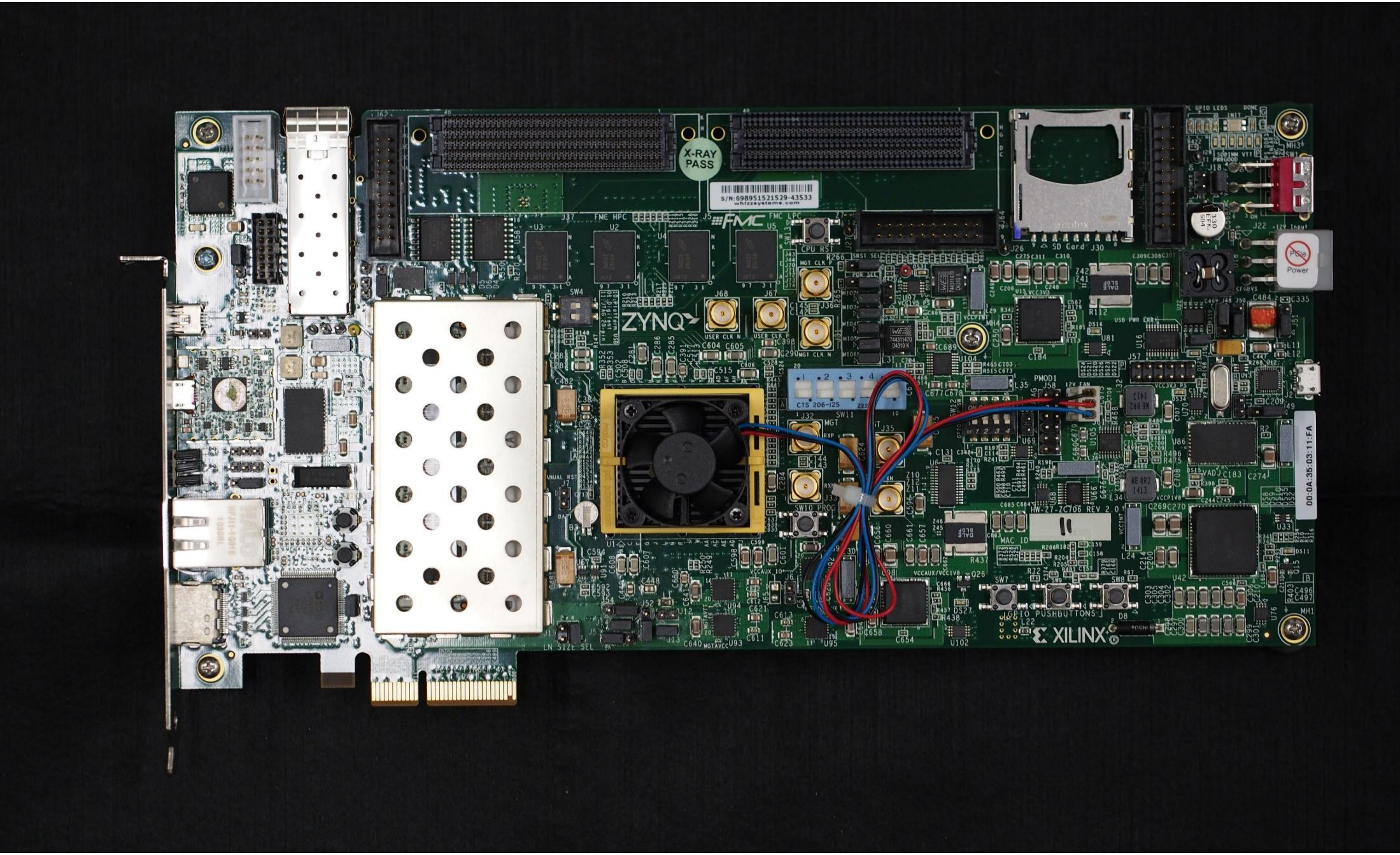
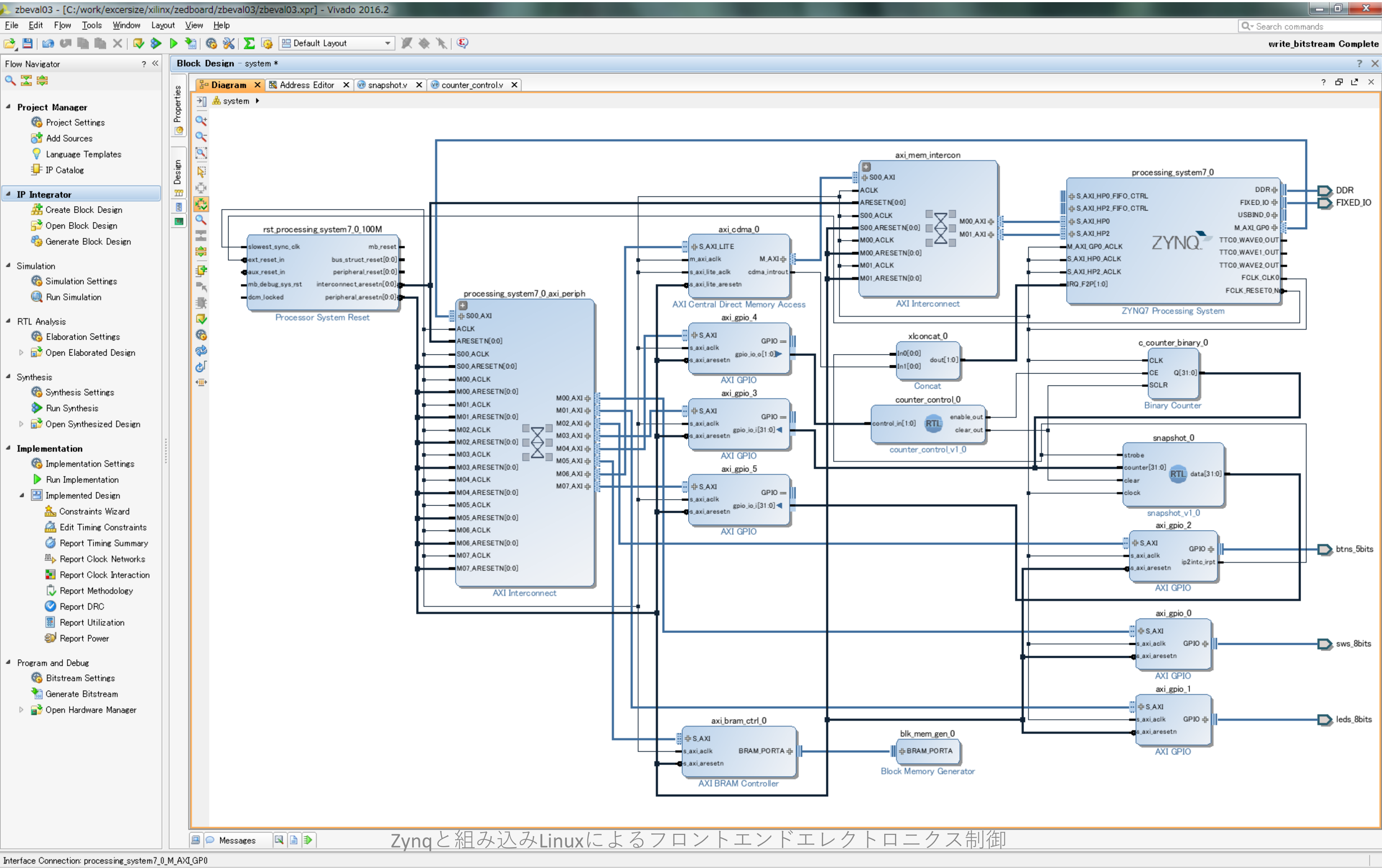


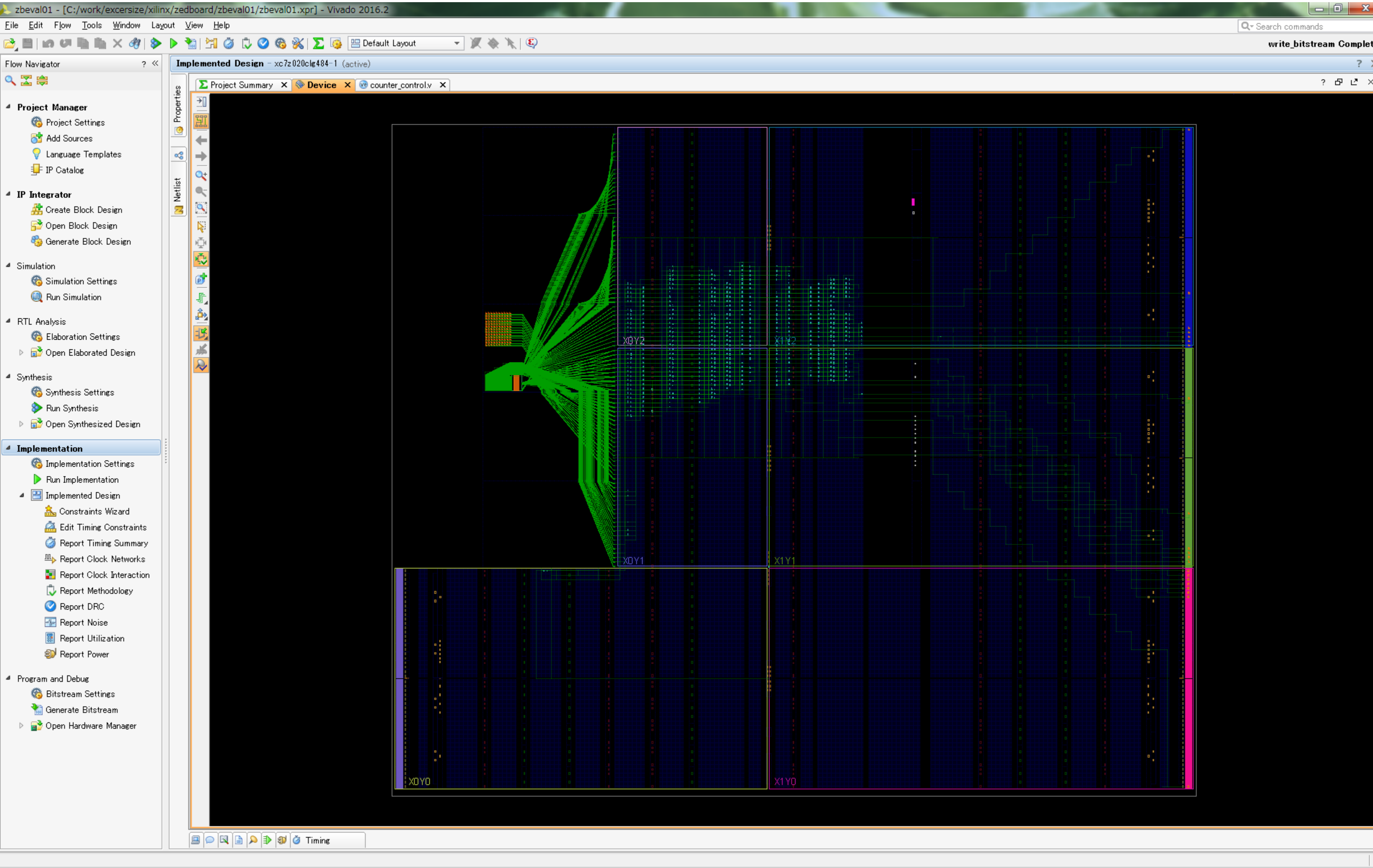
Figure 1 – ZedBoard Block Diagram

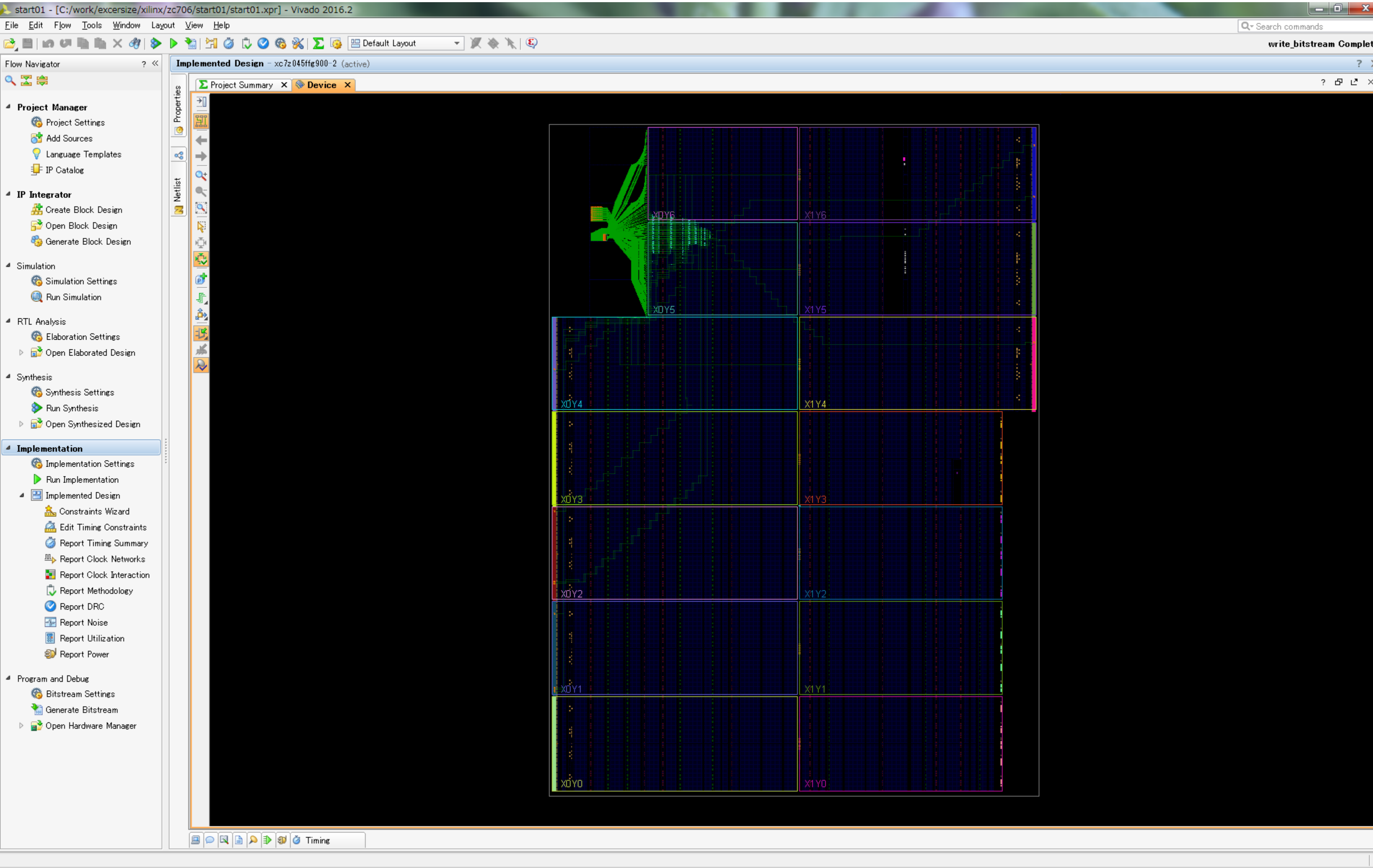
ZedBoard Hardware User's Guide, Version 2.2 27 January 2014, AVNET


ZedBoard
www.zedboard.org
DIGILENT
www.digilentinc.com
Copyright 2012
AVNET
LINE IN
MIC
LINE OUT
HPH OUT
HDMI OUT
XADC HEADER
JTAG
POWER
PROG
PS-RST
DONE
PB200-248
REV D
QC.OK
STM-5
94V-0


X-RAY PASS
FMC
ZYNQ
XILINX

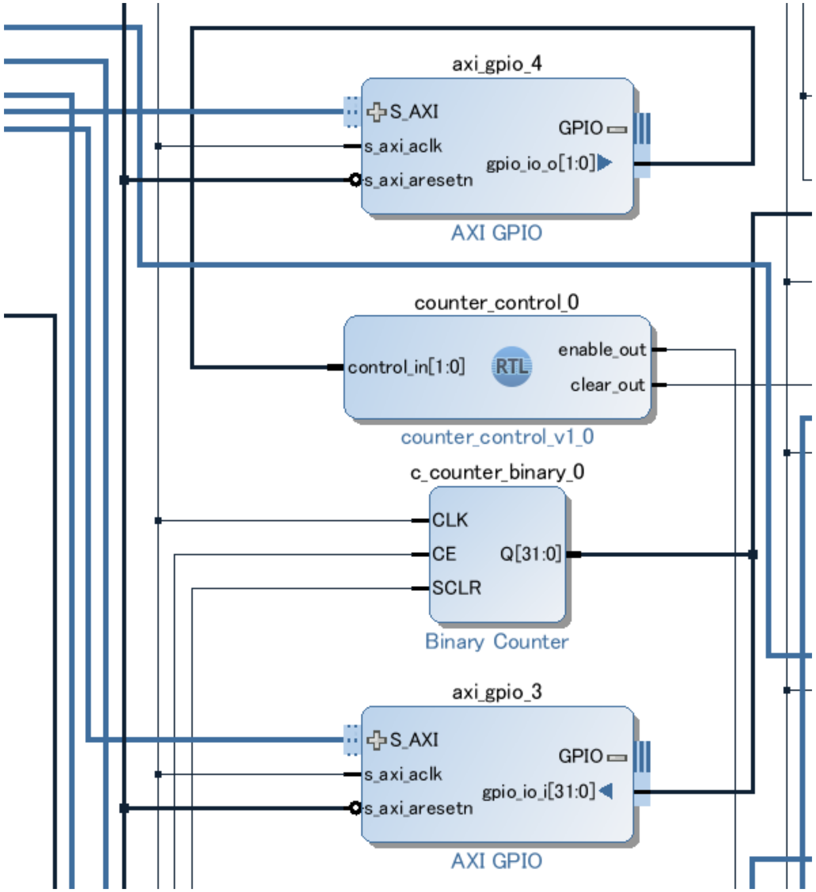

zbeval03 - [C:/work/excersize/xilinx/zedboard/zbeval03/zbeval03.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Default Layout
write_bitstream Complete
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Run Implementation
Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Utilization
Report Power
Program and Debug
Bitstream Settings
Generate Bitstream
Open Hardware Manager
Block Design - system *
Diagram
Address Editor
snapshot.v
counter_control.v
system
rst_processing_system7_0_100M
Processor System Reset
processing_system7_0_axi_periph
AXI Interconnect
axi_mem_intercon
AXI Interconnect
axi_cdma_0
AXI Central Direct Memory Access
processing_system7_0
ZYNQ
ZYNQ7 Processing System
DDR
FIXED_IO
axi_gpio_4
AXI GPIO
xlconcat_0
Concat
c_counter_binary_0
Binary Counter
axi_gpio_3
AXI GPIO
counter_control_0
counter_control_v1_0
axi_gpio_5
AXI GPIO
snapshot_0
snapshot_v1_0
axi_gpio_2
AXI GPIO
btns_5bits
axi_gpio_0
AXI GPIO
sws_8bits
axi_gpio_1
AXI GPIO
leds_8bits
axi_bram_ctrl_0
AXI BRAM Controller
blk_mem_gen_0
Block Memory Generator
Messages
Interface Connection: processing_system7_0_M_AXI_GP0


zbeval01 - [C:/work/excersize/xilinx/zedboard/zbeval01/zbeval01.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Default Layout
Search commands
write_bitstream Complete
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Run Implementation
Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Noise
Report Utilization
Report Power
Program and Debug
Bitstream Settings
Generate Bitstream
Open Hardware Manager
Implemented Design - xc7z020clg484-1 (active)
Project Summary
Device
counter_control.v
Properties
Netlist
X0Y2
X1Y2
X0Y1
X1Y1
X0Y0
X1Y0
Timing


start01 - [C:/work/excersize/xilinx/zc706/start01/start01.xpr] - Vivado 2016.2
File Edit Flow Tools Window Layout View Help
Default Layout
Search commands
write_bitstream Complete
Flow Navigator
Project Manager
Project Settings
Add Sources
Language Templates
IP Catalog
IP Integrator
Create Block Design
Open Block Design
Generate Block Design
Simulation
Simulation Settings
Run Simulation
RTL Analysis
Elaboration Settings
Open Elaborated Design
Synthesis
Synthesis Settings
Run Synthesis
Open Synthesized Design
Implementation
Implementation Settings
Run Implementation
Implemented Design
Constraints Wizard
Edit Timing Constraints
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Noise
Report Utilization
Report Power
Program and Debug
Bitstream Settings
Generate Bitstream
Open Hardware Manager
Implemented Design - xc7z045ffg900-2 (active)
Project Summary
Device
Properties
Netlist
X0Y6
X1Y6
X0Y5
X1Y5
X0Y4
X1Y4
X0Y3
X1Y3
X0Y2
X1Y2
X0Y1
X1Y1
X0Y0
X1Y0
Timing

計数の方法

- 32ビットバイナリーカウンター
- システムクロックでの駆動(100MHz)
- 10ns分解能
- GPIOで読み書き



ベアメタルプログラム～GPIO

変数代入235ns
ポインター代入241ns

- 方法1

  XGpio_DiscreteWrite(&CounterControlOutput, COUNTER_CONTROL_CHANNEL, COUNTER_ENABLE_BIT);

  cvalue = XGpio_DiscreteRead(&CounterDataInput, COUNTER_DATA_CHANNEL);

- 方法2

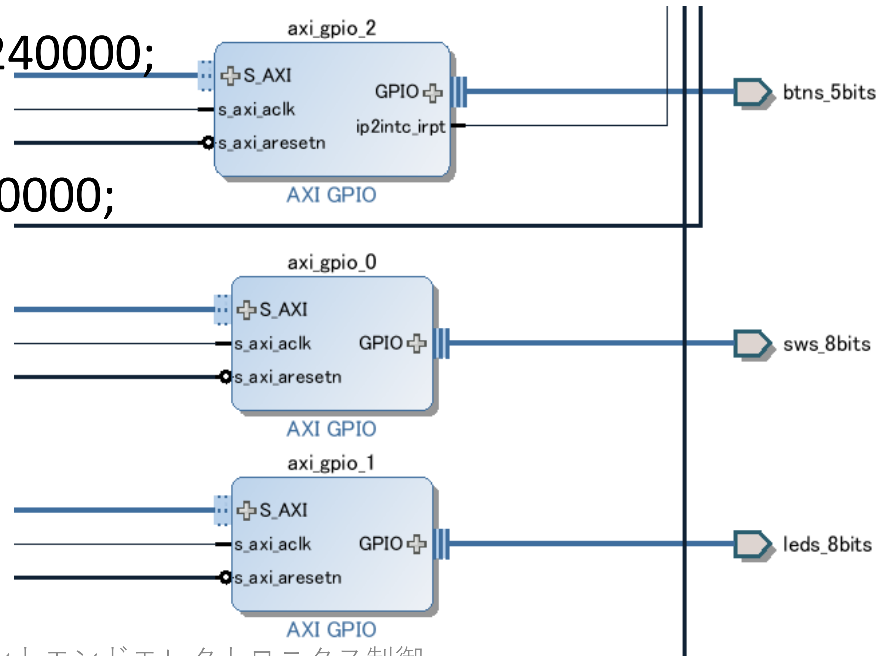
  unsigned char * timer_control = 0x41240000;

  * timer_control = 0x02;

  unsigned long * timer_value = 0x41230000;

  Unsigned long lap = * timer_value;

ポインター代入226ns



m_virtual_address = (unsigned int*) m_virtual_base + (m_physical_address − m_address_offset);

Linuxプログラム～GPIO

```
unsigned int m_physical_address;
::off_t m_address_offset;
unsigned int * m_virtual_base;
unsigned int * m_virtual_address;
int m_file_descriptor;

m_physical_address = 0x41200000;
m_file_descriptor = open( "/dev/mem", O_RDWR | O_SYNC );
m_address_offset = m_physical_address & ~(getpagesize()-1);
m_virtual_base = (unsigned int*)mmap(NULL, 1, PROT_READ | PROT_WRITE,
            MAP_SHARED, m_file_descriptor, m_address_offset );
m_virtual_address = (unsigned int*) m_virtual_base +
            ( m_physical_address - m_address_offset );
Int cvalue = * m_virtual_address;
```

ポインター代入250ns

BRAM

ベアメタルの場合

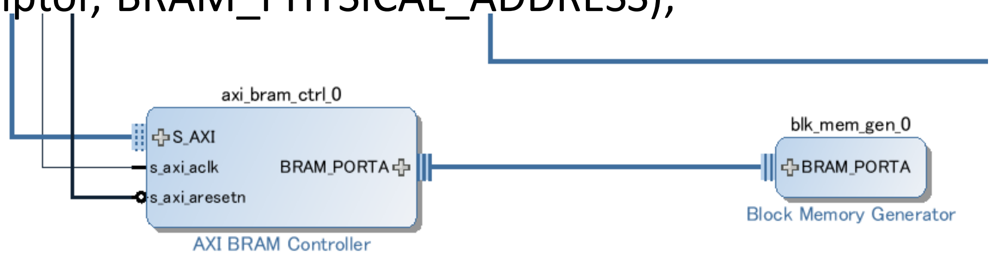
```
unsigned int * uiDataArray = (unsigned int *)
        XPAR_AXI_BRAM_CTRL_0_S_AXI_BASEADDR;
Int uiLoop, uiData;
*(unsigned int *)(uiDataArray + uiLoop) = uiLoop;
uiData = *(unsigned int *)(uiDataArray + uiLoop);
```

Write: 158ns
Read: 190ns

Linuxの場合

```
enum { BRAM_PHYSICAL_ADDRESS = 0x40000000,
        BRAM_MEMORY_SIZE = 0x2000};
m_file_descriptor = open( "/dev/mem", O_RDWR | O_SYNC );
m_virtual_base = (unsigned int*)mmap(NULL, 1, PROT_READ | PROT_WRITE,
        MAP_SHARED, m_file_descriptor, BRAM_PHYSICAL_ADDRESS);
```

Write: 112ns
Read: 207ns
(Cf. DDR: 21ns)



Measured Access Speed for GPIO and BRAM

Platform	Operation	Implementation	Board	Speed
Bare Metal	GPIO Read	*DDRPointer = XGpio_DiscreteRead();	ZedBoard	241ns
	GPIO Read	*DDRPointer = XGpio_DiscreteRead();	ZC706*	420ns
	GPIO Read	DDRData = XGpio_DiscreteRead();	ZedBoard	235ns
	GPIO Read	*DDRPointer = * DirectPointer;	ZedBoard	226ns
	BRAM Write	* DirectPointer = * DDRPointer;	ZedBoard	158ns
	BRAM Read	* DDRPointer = * DirectPointer;	ZedBoard	190ns
Linux	GPIO Read	* DDRPointer = * MmapPointer;	ZedBoard	250ns
	BRAM Write	* MMapPointer = * DDRPointer;	ZedBoard	112ns
	BRAM Read	* DDRPointer = * MMapPointer;	ZedBoard	207ns
	DDR Read	* DstDDRPointer = * SrcDDRPointer;	ZedBoard	21ns

*) PL clock frequency of ZC706 was 50MHz which is the half of ZedBoard (100MHz)

Dhrystone benchmark: 1,108,060 (= 630 DMIPS, 0.95 DMIPS/MHz) for both boards.

ベアメタルプログラム～割り込み

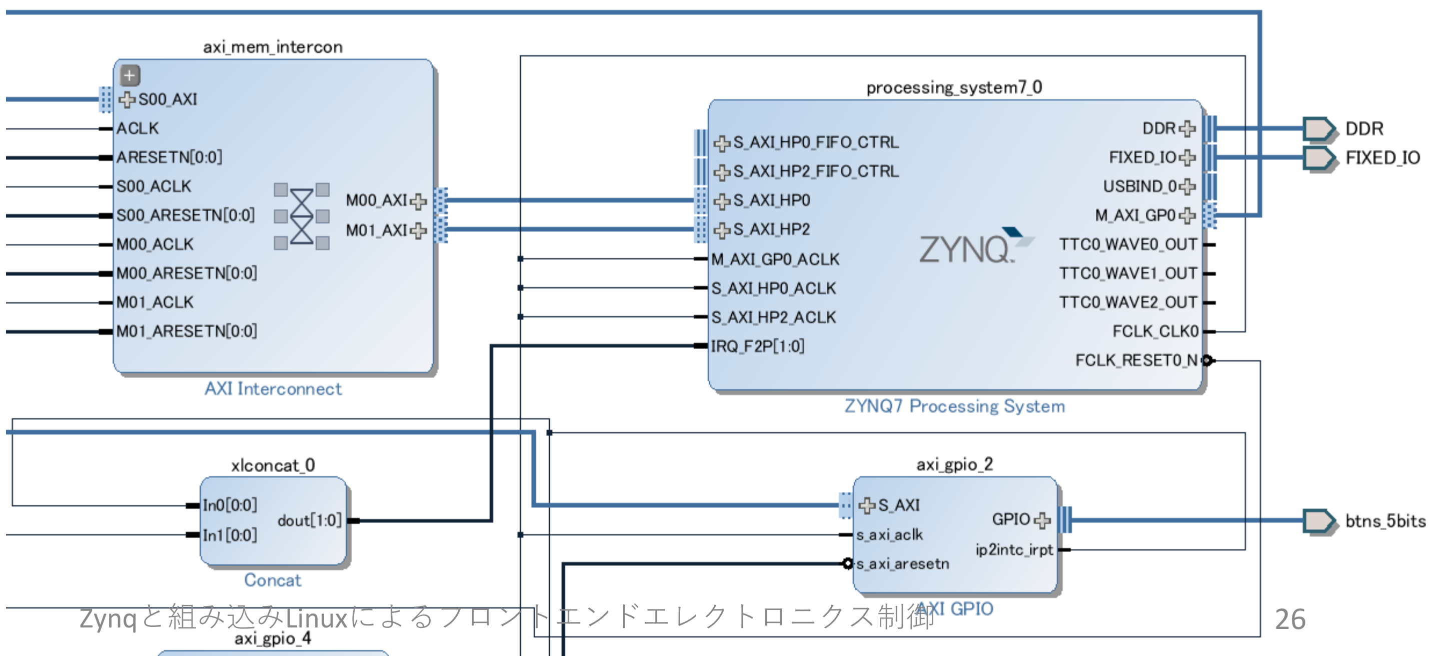
```
status = XScuGic_Connect(&INTCInst, INTC_GPIO_INTERRUPT_ID,
(Xil_ExceptionHandler)BTN_Intr_Handler, (void *)GpioInstancePtr);

XGpio_InterruptEnable(GpioInstancePtr, 1);

XGpio_InterruptGlobalEnable(GpioInstancePtr);

XScuGic_Enable(&INTCInst, INTC_GPIO_INTERRUPT_ID);
```

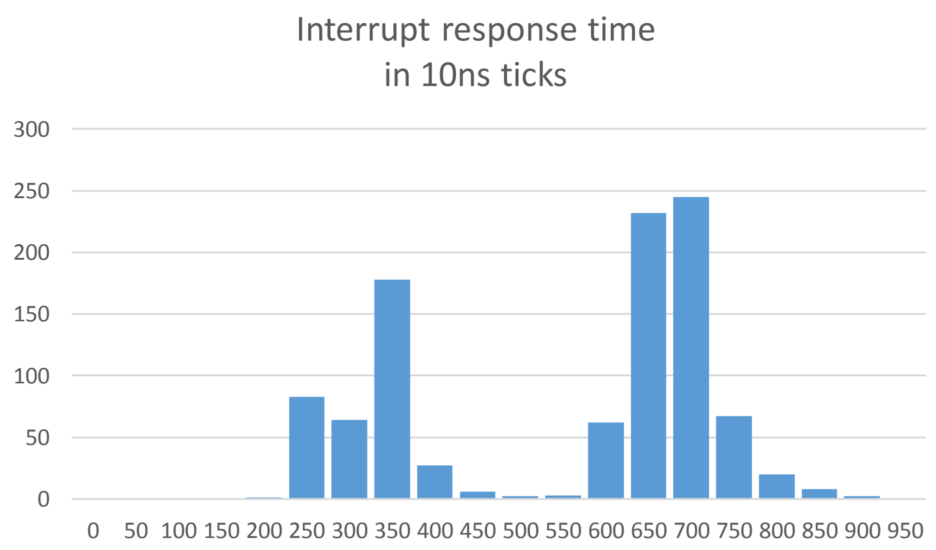
割り込み事象が発生してから
ハンドラーに飛ぶまでの時間
初回　730ns　次回以降　220ns



Linuxプログラム～割り込み

- 基本的にカーネルモジュールで操作する。
- uio(User Space IO)を用いてユーザスペースから操作が可能だが新しいリリースではサポートされていない。
- インタラプトハンドラーをカーネルモジュールとして作成
  - カーネルバージョン4.4
  - Device Treeによるハードウエア記述
  - GPIO consumerとしてデザインする

比較的大きなレスポンスタイム(スイッチが押されてからハンドラーが呼ばれるまでの時間)
3.5µsと7.0µsにピーク



Linuxプログラム～/sys/class/gpio

```
fd_export = open("/sys/class/gpio/export", O_WRONLY);
write( fd_export, “901”, 3);
close( fd_export );
fd_switch_value= open("/sys/class/gpio/gpio901/value", O_RDONLY);
char switch_value[2];
read( fd_switch_value, &switch_value, 1 );
close( fd_switch_value );
fd_unexport = open("/sys/class/gpio/unexport", O_WRONLY );
write( fd_unexport, “901”, 3);
close( fd_unexport );
```

オーバーヘッドが極めて大きく、1ビットのアクセスに4,510nsかかっている。実用的ではない。
/sys/classはユーザスペースからデバッグ等で利用するもの。

Dhrystone Benchmark

- 得られた結果はZedBoardの場合937,448。Wikipediaによるとこの数字を1757(VAX11/780のDhrystone値)で割ったものをDMIPSというらしいが、この場合534になる。ZedBoardでは666.7MHzでARMが動いている。この結果0.800DMIPS/MHz。
- ARMの公称値はCortex-A9 2.5 DMIPS/MHzだそうだ(伝聞)半分も出ていないがいいのか？
- Linuxで動かしてみた値は943,396.2。これだと0.805DMIPS/MHz。

開発計画

- 2016年度
  - 評価ボードによる技術検証
    - 割り込み・DMA
    - GTX等のソフトウエア制御
    - Partial Reconfiguration
    - HLS (Higher Level Synthesis)
  - Linuxによる制御技術の確立
    - カーネルハッキング
    - デバイスドライバー
    - サポートライブラリ
- 2017年度
  - PTZ仕様決定
  - PTZデザイン及び試作
- 2018年度
  - PTZ量産
  - フロントエンドエレクトロニクス開発デモ

まとめに代えて

- SoCデバイスの有効性
  - 大規模アプリケーションで特に有効
  - 開発サイクルの大幅短縮
  - 高度な診断・モニターが可能
- 広範囲にわたる開発課題
  - ハードウエア
    - IPベースの開発・IPパッケージ化
    - HLSの採用・最適化技術
  - ソフトウエア
    - カーネルモジュール開発とOSの構築
    - サポートツール開発
- 共同開発の必要性
  - IPやソフトウエアなど産物の共有
  - OpenItプロジェクト