2023年度(令和5年)版

Ver. 2023-10-02a

Course number: CSC.T363

コンピュータアーキテクチャ
Computer Architecture

1. コンピュータの構成と動作原理
Guidance and Fundamentals of Computer Design

www.arch.cs.titech.ac.jp/lecture/CA/
Tue 13:30-15:10, 15:25-17:05
Fri 13:30-15:10

吉瀬 謙二 情報工学系
Kenji Kise, Department of Computer Science
kise _at_ c.titech.ac.jp

お知らせ

- 講義および演習は、情報工学系計算機室で実施します。
  - 情報工学系計算機室に集まってください。http://www.csc.titech.ac.jp/
- 講義の最新情報は次のサポートページを確認してください。
  - www.arch.cs.titech.ac.jp/lecture/CA/

arch.cs.titech.ac.jp/lecture/CA/

Computer Architecture (CSC.T363) Support Page, Dept. of Computer Science TOKYO TECH

News

- This page was refreshed for the new quarter. (2023-10-03)
- This page was refreshed for the new quarter. (2022-09-23)
- This page was refreshed for the new quarter. (2021-09-30)
- This page is open. (2018-09-27)

Final Exam

- 2023-12-01 (13:30-15:10) final exam in the computer room of the department of computer science

Materials for Computer Architecture Exercise

Lectures and exercises will be held in the computer room of the department of computer science.

- Exercise_1 2023-10-10 (15:25-17:05): Exercise(1) (UART, Input/Output)
- Exercise_2 2023-10-17 (15:25-17:05): Exercise(2)

【重要】ACRiルームのアカウント

- Computer Logic Design の講義で使ったアカウントがあれば、そのアカウントを用いる。

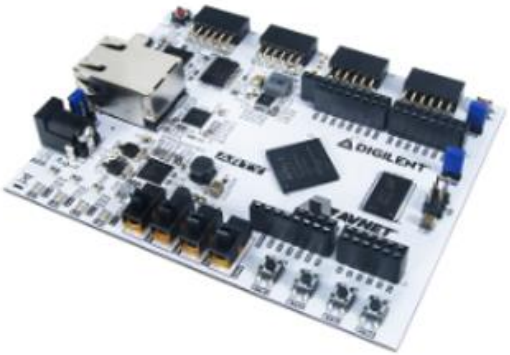
- アカウントがなければ、今、次のURLのページを参考にアカウントを申請すること。
  - https://gw.acri.c.titech.ac.jp/wp/manual/apply-for-account

【重要】情報工学系計算機室のアカウント

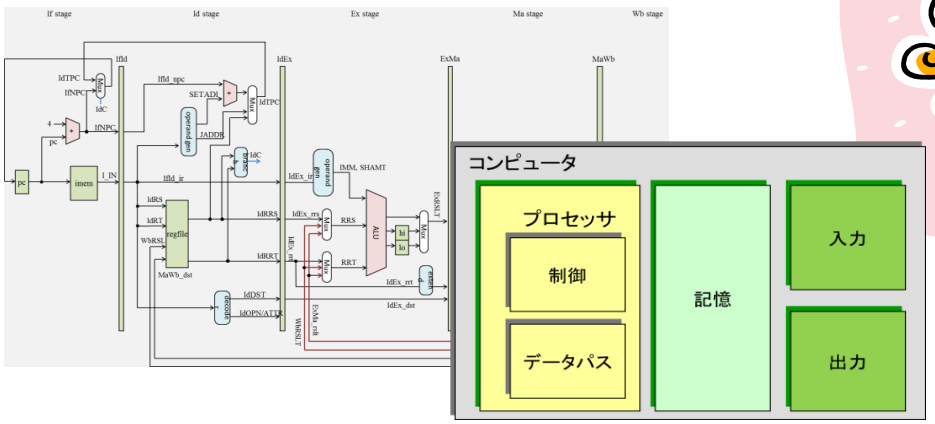
- 2021年4月より新システムに移行しているので、旧システムでアカウントを発行済みの場合でもパスワード発行が必要。
- 目の前のコンピュータにログインできない場合には、今、アカウントを作成する。

この講義の特徴

- 講義2単位，演習1単位.
- 演習は前半がハンズオン形式、後半がグループでの作業
- 1人1台のFPGA (Field-Programmable Gate Array) ボードを用いた演習.
  - 搭載するFPGA: XC7A35TICSG324-1L
- 教科書で説明されるRISC-Vのコンピュータを ハードウェア記述言語Verilog HDLで記述し，FPGAボードに実装する.
- コンピュータの高速化に取り組み，コンテスト形式で成果を競う.
  - コンテストでは、グループではなく個人の成果を発表。

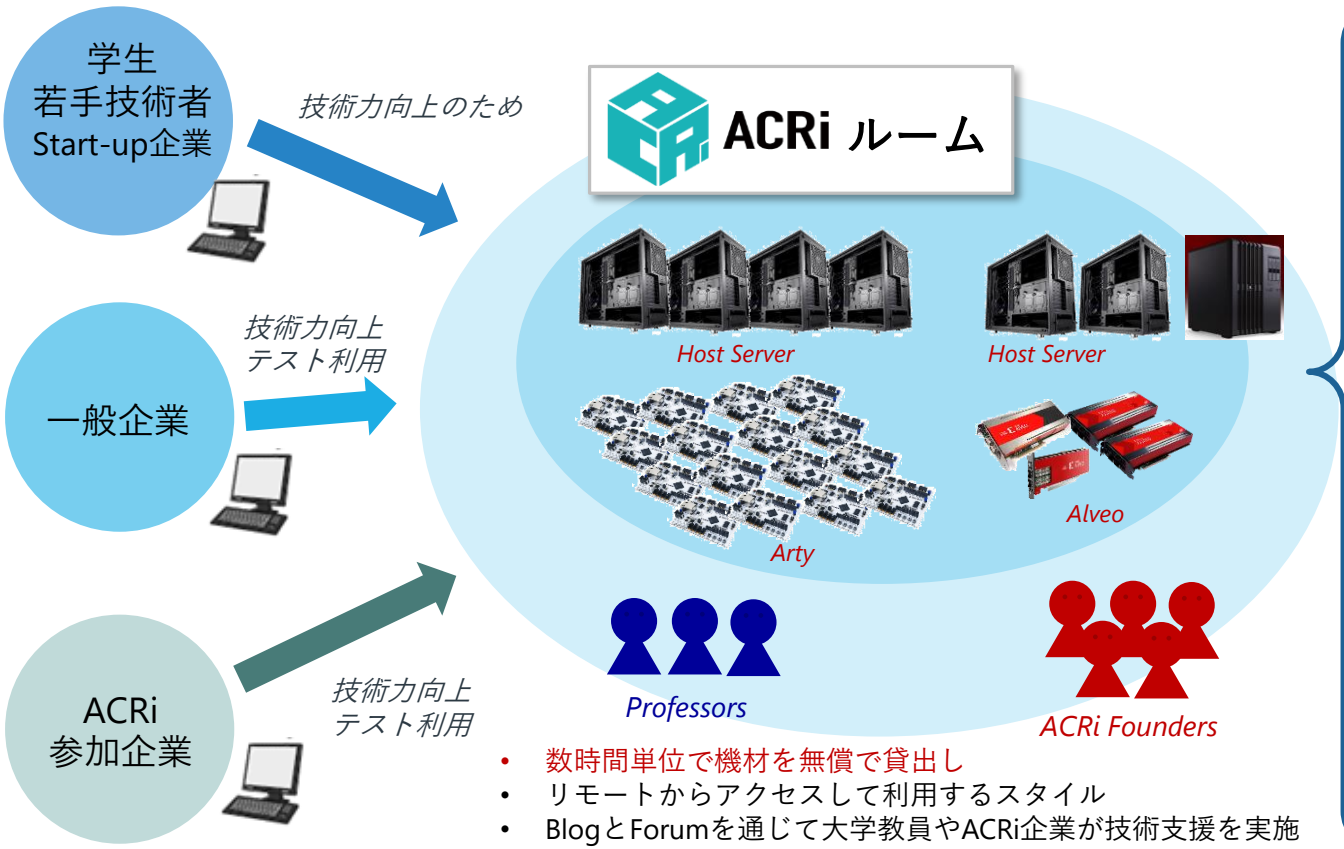


```
module main (clk, led);
  input  wire clk;
  output wire led;

  reg [26:0] cnt;
  always @(posedge clk) cnt <= cnt + 1;

  assign led = cnt[26];
endmodule
```



ACRiルーム (FPGA利用環境)

－ 日本初、産学連携でFPGA検証環境と学習機会を無償で提供 －



- 数時間単位で機材を無償で貸出し
- リモートからアクセスして利用するスタイル
- BlogとForumを通じて大学教員やACRi企業が技術支援を実施

FPGA Server

- CPU: Core i9 (8 core /16 thread)
- メモリ: DDR4 128GB (32GB x 4)
- ストレージ: SSD M.2 1TB x 2

Arty A7-35T

- 1サーバにArtyを15枚設置
- Digilent Arty A7-35T
- ユーザ毎にVMを割り当て
- 3時間でVMを再起動

Alveo

- 1サーバにAlveoを1枚設置
- Xilinx Alveo
  U50 / U200 / U250 / U280

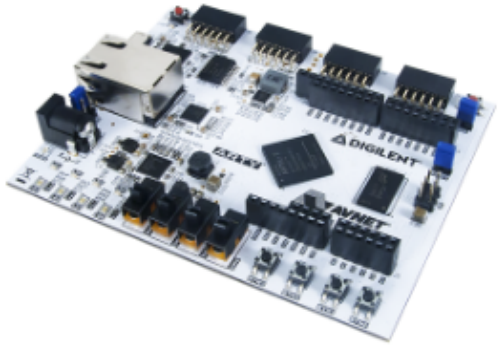
アダプティブコンピューティング研究推進体
Adaptive Computing Research Initiative (ACRi)

www.acri.c.titech.ac.jp

Digilent Arty A7-35T

Arty A7

The Arty A7, formerly known as the Arty, is a ready-to-use development platform designed around the Artix-7™ Field Programmable Gate Array (FPGA) from Xilinx. It was designed specifically for use as a MicroBlaze Soft Processing System. When used in this context, the Arty A7 becomes the most flexible processing platform you could hope to add to your collection, capable of adapting to whatever your project requires. Unlike other Single Board Computers, the Arty A7 isn't bound to a single set of processing peripherals: One moment it's a communication powerhouse chock-full of UARTs, SPIs, IICs, and an Ethernet MAC, and the next it's a meticulous timekeeper with a dozen 32-bit timers.



搭載するFPGA: XC7A35TICSG324-1L

Store

Reference Manual | Technical Support

Arty A7

Artix-7 FPGA Development Board

Features

- Programmable over JTAG and Quad-SPI Flash
- On-chip analog-to-digital converter

Key Specifications

FPGA Part #	XC7A35TICSG324-1L (XC7A100TCSG324-1*)
Logic Slices	5,200 (15,850*)
Block RAM	1,800 Kbits (4,860* Kbits)
DSP Slices	90 (240*)
DDR3	256 MB @ 667 MHz
Internal clock	450 MHz+
Quad-SPI Flash	16 MB
Ethernet	10/100 Mbps

https://reference.digilentinc.com/reference/programmable-logic/arty-a7/start

Syllabus (1/3)

講義の概要とねらい

情報工学分野において、コンピュータの動作原理と高性能化のための方式を理解し、さらにハードウェアとソフトウェアの相互関係を習得することは非常に重要です。本講義では、プロセッサ、メモリ、入出力、マルチプロセッサ、マルチコアというコンピュータシステムを構成する各種装置について、その役割と動作原理を学びます。
演習では、Verilog HDL等のハードウェア記述言語を用いてFPGAが搭載されたハードウェアボード等に高度なディジタル回路であるプロセッサを実装し、プログラムを動作させることでハードウェアとソフトウェアの相互関係を習得します。

到達目標

本講義を履修することによってコンピュータ以下を習得する。
・コンピュータの動作原理と高性能化のためのアーキテクチャ
・プロセッサ、メモリ、入出力、マルチプロセッサ、マルチコアといったコンピュータシステムを構成する各種装置の役割と動作原理
・ハードウェア記述言語を用いた一般的なコンピュータシステムの設計能力

キーワード

コンピュータアーキテクチャ、プロセッサ、スーパースカラ、キャッシュ、メモリ階層、仮想記憶、マルチプロセッサ、マルチコア、ハードウェア記述言語、Verilog HDL、FPGA

学生が身につける力(ディグリー・ポリシー)

✓ 専門力	教養力	コミュニケーション力	展開力(探究力又は設定力)	✓ 展開力(実践力又は解決力)

授業の進め方

原則として、100分×2コマの講義の後、100分×1コマのFPGAボードを用いた演習をおこないます。

Syllabus (2/3)

授業計画・課題		
	授業計画	課題
第1回	コンピュータの構成と動作原理	コンピュータの構成と動作原理について理解する。
第2回	コンピュータの性能と消費電力の動向	コンピュータの性能と消費電力の動向について理解する。
第3回	コンピュータ設計演習(1)	コンピュータ設計演習(1)
第4回	半導体メモリ	半導体メモリについて理解する。
第5回	コンピュータ設計演習(2)	コンピュータ設計演習(2)
第6回	キャッシュ：ダイレクトマップ方式	ダイレクトマップ方式のキャッシュについて理解する。
第7回	キャッシュ：セットアソシアティブ方式	セットアソシアティブ方式のキャッシュについて理解する。
第8回	コンピュータ設計演習(3)	コンピュータ設計演習(3)
第9回	メモリシステムの階層化と信頼性	メモリシステムの階層化と信頼性について理解する。
第10回	パイプラインプロセッサ	パイプラインプロセッサについて理解する。
第11回	コンピュータ設計演習(4)	コンピュータ設計演習(4)
第12回	スーパースカラプロセッサ	スーパースカラプロセッサについて理解する。
第13回	分岐予測	分岐予測について理解する。
第14回	コンピュータ設計演習(5)	コンピュータ設計演習(5)
第15回	ベクタ、SIMDにおけるデータレベル並列性	ベクタ、SIMDにおけるデータレベル並列性について理解する。
第16回	仮想記憶、セキュリティ	仮想記憶、セキュリティについて理解する。
第17回	コンピュータ設計演習(6)	コンピュータ設計演習(6)
第18回	入出力、バス	入出力、バスについて理解する。
第19回	相互接続ネットワーク	相互接続ネットワークについて理解する。
第20回	コンピュータ設計演習(7)	コンピュータ設計演習(7)
第21回	マルチプロセッサ、マルチコア	マルチプロセッサ、マルチコアについて理解する。

最新の情報はサポートページに掲載する。

Syllabus (3/3)

教科書
デイビッド・A. パターソン、ジョン・L. ヘネシー (著)、成田光彰 (翻訳)『コンピュータの構成と設計 第5版 上/下』日経BP社
参考書、講義資料等
無し。
成績評価の基準及び方法
講義で扱うコンピュータアーキテクチャに関する理解、ハードウェア記述言語を用いたコンピュータシステム実装への応用力を評価する。演習（60%）と期末試験（40%）により評価する。
関連する科目
CSC.T252 ： 論理回路理論 CSC.T262 ： アセンブリ言語 CSC.T372 ： コンパイラ構成 CSC.T341 ： コンピュータ論理設計 CSC.T433 ： 先端コンピュータアーキテクチャ
履修の条件(知識・技能・履修済科目等)
履修条件は特に設けないが、関連する科目のコンピュータ論理設計を履修していることが望ましい。

講義日程(予定)

Lectures and exercises will be held in the computer room of the department of computer science.

- Exercise_1 2023-10-10 (15:25-17:05): Exercise(1) (UART, Input/Output)
- Exercise_2 2023-10-17 (15:25-17:05): Exercise(2)
- Exercise_3 2023-10-24 (15:25-17:05): Exercise(3)
- Exercise_4 2023-11-07 (15:25-17:05): Exercise(4)
- Exercise_5 2023-11-14 (15:25-17:05): Exercise(5)
- Exercise_6 2023-11-21 (15:25-17:05): Exercise(6)
- Exercise_7 2023-11-28 (13:45-17:20): Computer Design Contest

---

Lecture Slides and Materials

Lectures and exercises will be held in the computer room of the department of computer science.

- Lecture_01 2023-10-03 (13:30-15:10): Guidance and Fundamentals of Computer Design
- Lecture_02 2023-10-06 (13:30-15:10):
- Lecture_03 2023-10-10 (13:30-15:10):
- Lecture_04 2023-10-13 (13:30-15:10):
- Lecture_05 2023-10-17 (13:30-15:10):
- Lecture_06 2023-10-20 (13:30-15:10):
- Lecture_07 2023-10-24 (13:30-15:10):
- Lecture_08 2023-10-31 (13:30-15:10):
- Lecture_09 2023-11-07 (13:30-15:10):
- Lecture_10 2023-11-10 (13:30-15:10):
- Lecture_00 2023-11-17 (13:30-15:10): canceled
- Lecture_11 2023-11-21 (13:30-15:10):
- Lecture_12 2023-11-24 (13:30-15:10):
- Lecture_13 2023-12-01 (13:30-15:10): Final Exam

教科書

コンピュータの構成と設計 第5版、パターソン＆ヘネシー
（成田光彰　訳）、日経BP社

アーキテクチャ(建築), Architecture

パルテノン神殿

世界最大のクフ王のピラミッド
1個約2.5tのブロックを 230～250万 個
積み重ねて造られている。

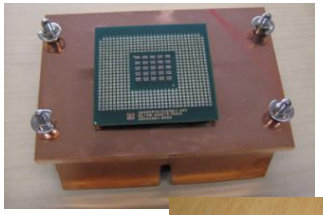
写真は計算機アーキテクチャのホームページから http://www.cs.wisc.edu/arch/www/

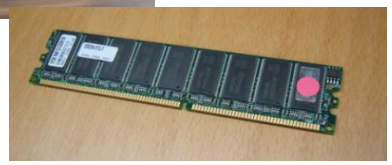
What's Computer Architecture?

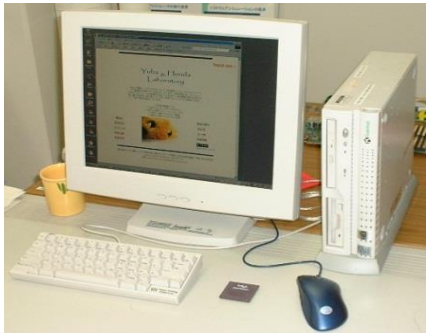
Computer Architecture is the science and art of selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals. Computer architecture is *not* about using computers to design buildings.

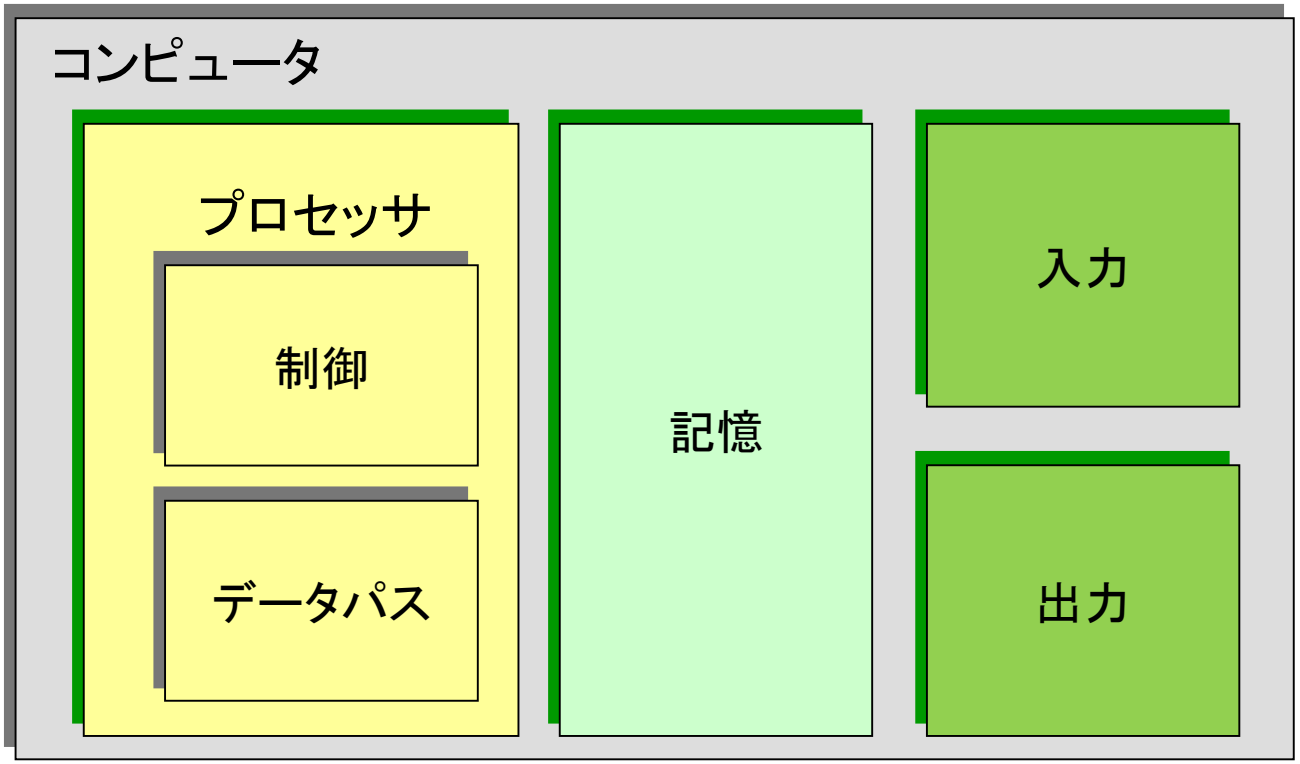
計算機アーキテクチャのホームページから http://www.cs.wisc.edu/arch/www/

コンピュータ(ハードウェア)の古典的な要素







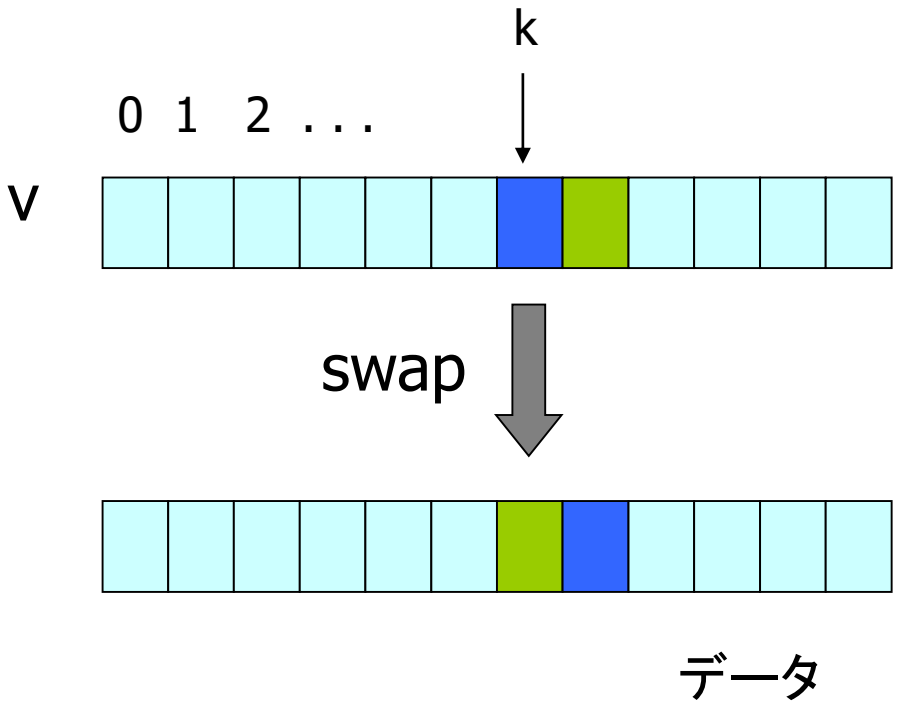


プロセッサが実行する小さい単位の処理である命令(machine instruction)を記憶装置から取り出して、その指示に従ってプロセッサに格納されているデータ、あるいは記憶装置から取り出したデータに対する演算をおこなう。また、得られた演算の結果はプロセッサの内部に格納したり、記憶装置に格納する。
コンピュータの外部からのデータを記憶装置に書き込むのが入力装置であり、記憶装置から読み出したデータを出力するのが出力装置である。記憶装置をメインメモリ、プロセッサをCPUと呼ぶことがある。

高水準言語からハードウェアの言語へ

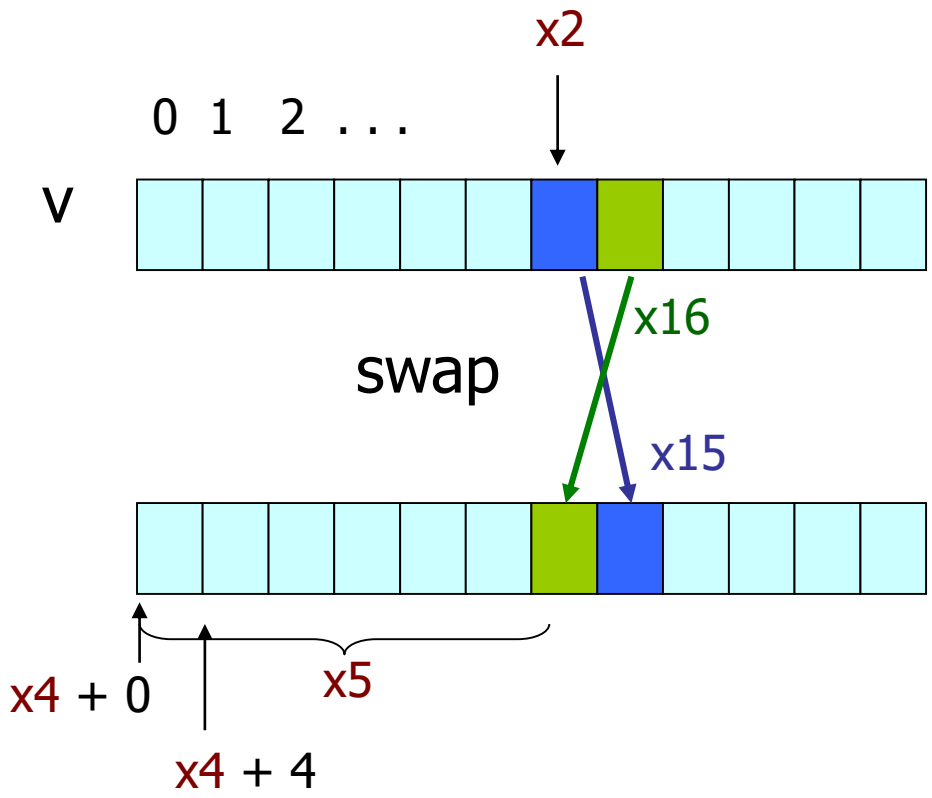
```
swap(int v[], int k)
{
    int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

C言語で記述したプログラム



高水準言語からハードウェアの言語へ

```
swap:
    add  x2, x5, x5
    add  x2, x2, x2
    add  x2, x4, x2
    lw   x15, 0(x2)
    lw   x16, 4(x2)
    sw   x16, 0(x2)
    sw   x15, 4(x2)
    ret
```

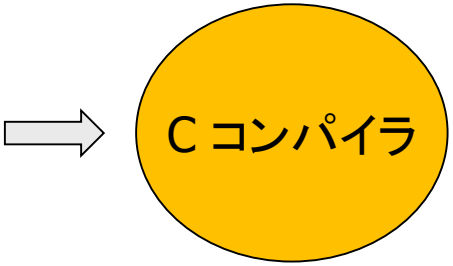


RISC-Vのアセンブリ言語に変換されたプログラム

高水準言語からハードウェアの言語へ

```
swap(int v[], int k)
{
    int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

C言語で記述したプログラム

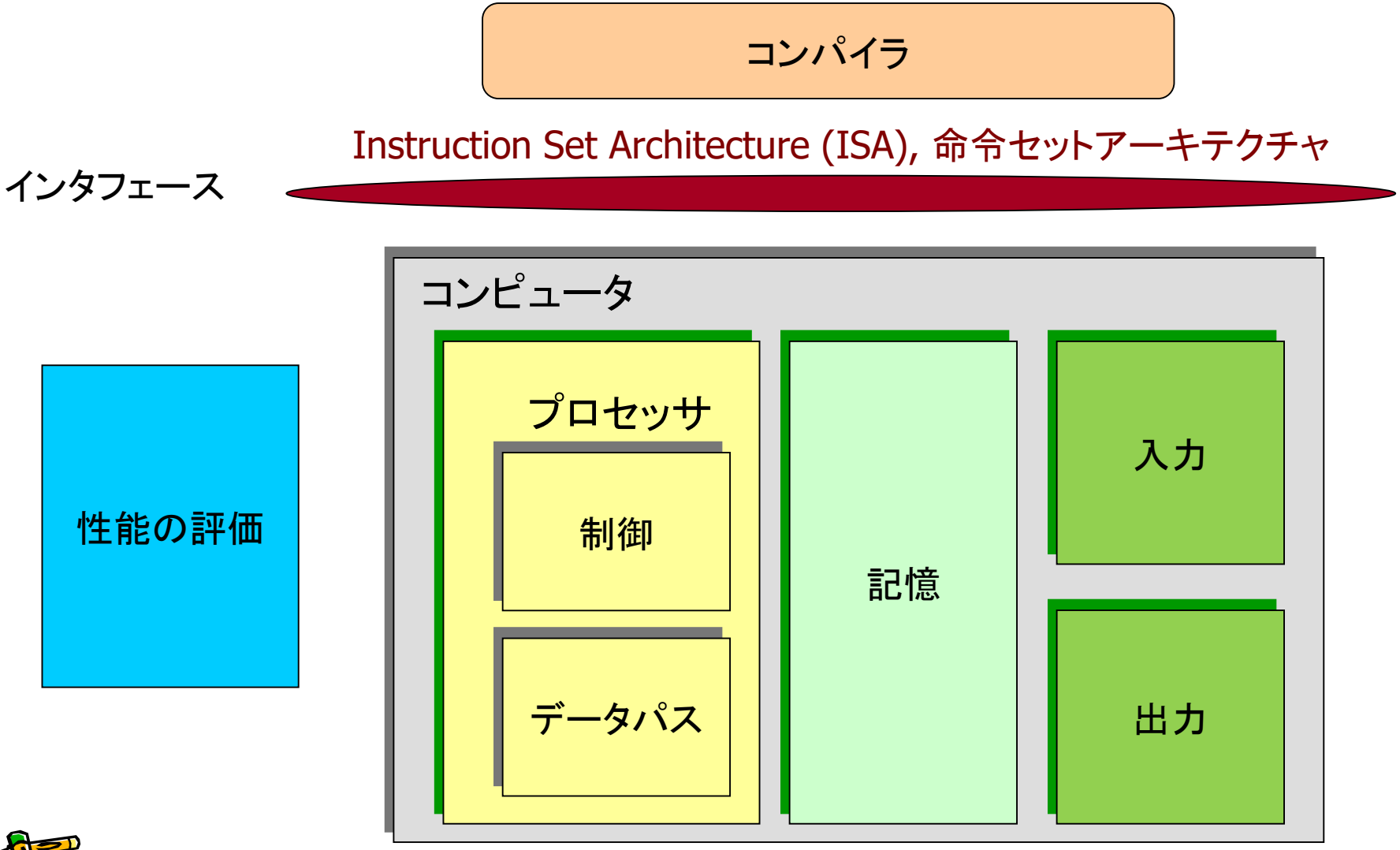



```
swap:
    add  x2, x5, x5
    add  x2, x2, x2
    add  x2, x4, x2
    lw   x15, 0(x2)
    lw   x16, 4(x2)
    sw   x16, 0(x2)
    sw   x15, 4(x2)
    ret
```

アセンブリ言語に変換された
プログラム

機械語に落とされたプログラム（機械命令の集まり）

コンピュータの古典的な要素

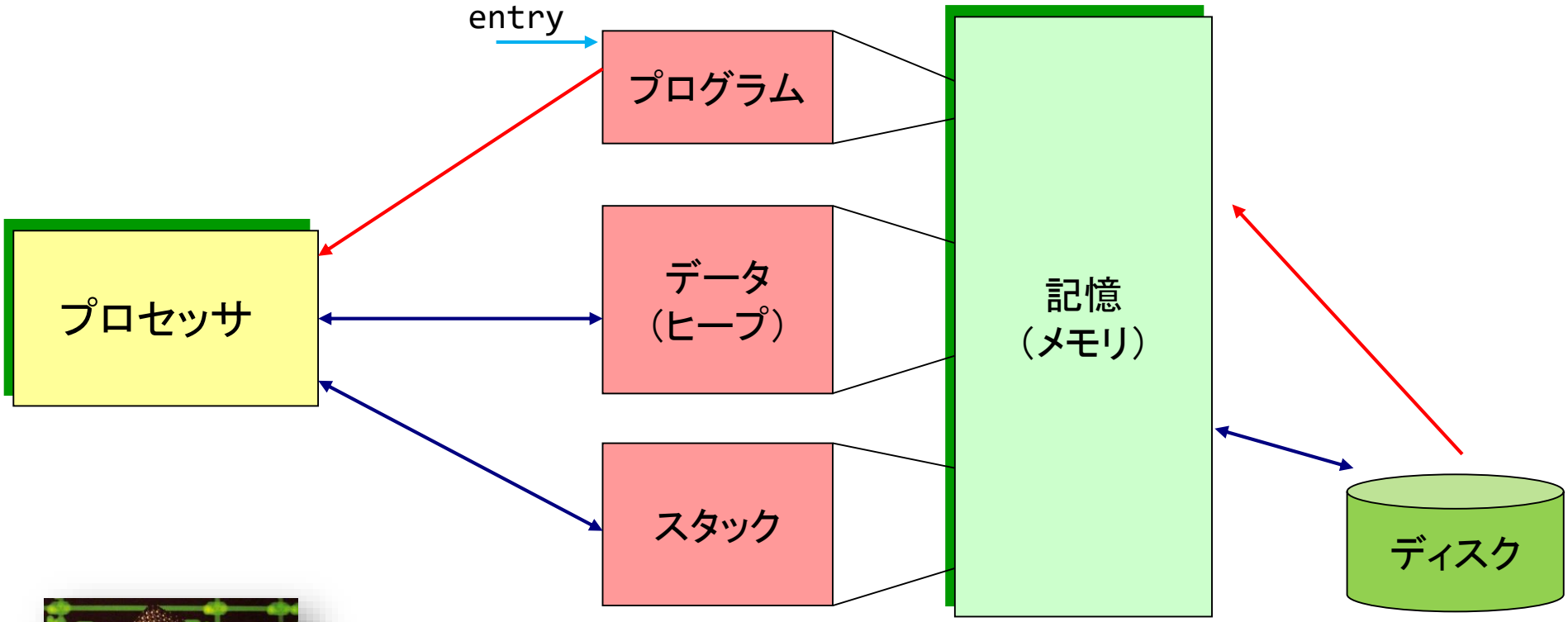


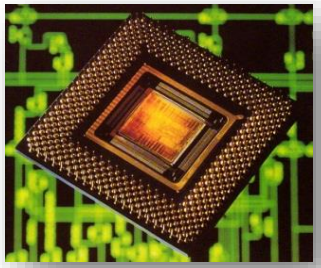
RISC-V Instruction Set Architecture (ISA)

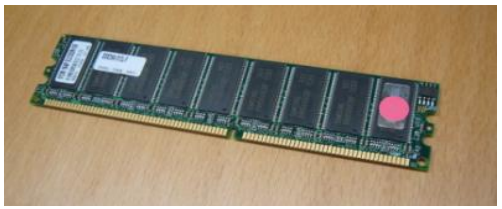
funct7	rs2	rs1	funct3	rd	opcode	R-type

0000000	rs2	rs1	000	rd	0110011	R-type	**add**
0100000	rs2	rs1	000	rd	0110011	R-type	**sub**
0000000	rs2	rs1	001	rd	0110011	R-type	**sll**
0000000	rs2	rs1	010	rd	0110011	R-type	**slt**
0000000	rs2	rs1	011	rd	0110011	R-type	**sltu**
0000000	rs2	rs1	100	rd	0110011	R-type	**xor**
0000000	rs2	rs1	101	rd	0110011	R-type	**srl**
0100000	rs2	rs1	101	rd	0110011	R-type	**sra**
0000000	rs2	rs1	110	rd	0110011	R-type	**or**
0000000	rs2	rs1	111	rd	0110011	R-type	**and**

プログラム，データ，その他







4GB (32bit) memory space and virtual memory

0x00000000

00000000 00000000 00000000 $00000000_2 = 0_{10}$

2GB Memory !

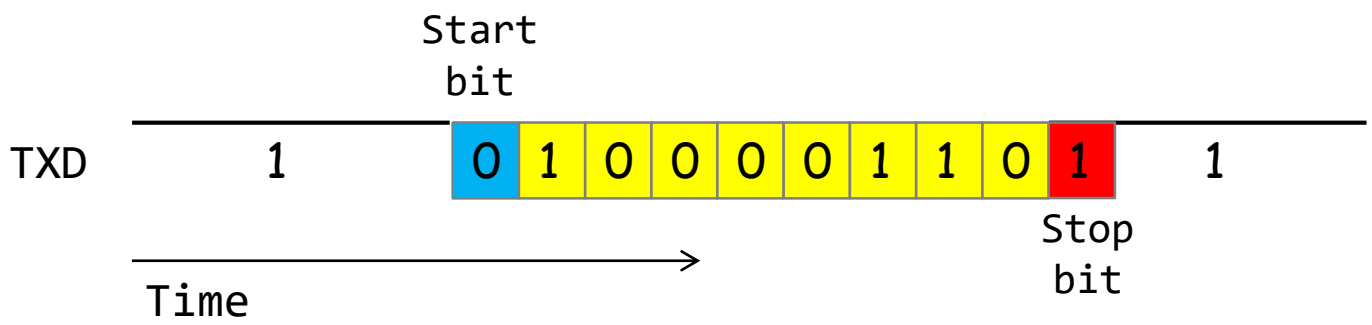
```
kterm
top - 11:35:26 up 10 days, 19:49,  2 users,  load average: 0.01, 0.01, 0
Tasks: 164 total,   1 running, 163 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,
Mem:   4002924k total,  3252404k used,   750520k free,   181808k buffers
Swap:  6062072k total,        0k used,  6062072k free,  2570804k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    1 root      15   0 10348  696  584 S  0.0  0.0   0:01.00 init
    2 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    4 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    5 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 migration/1
    6 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/1
    7 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/1
    8 root      RT  -5     0    0    0 S  0.0  0.0   0:00.01 migration/2
    9 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/2
   10 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/2
```

0xFFFFFFFF

11111111 11111111 11111111 $11111111_2 = 4{,}294{,}967{,}296 - 1_{10}$

UART (Universal Asynchronous Receiver/Transmitter)

- 調歩同期方式によるシリアル信号をパラレル信号に変換したり，その逆方向の変換をおこなう集積回路をUARTと呼ぶ．8ビット（1バイト）単位でデータを送信・受信する．
- UARTを用いることで，FPGAとコンピュータの間でのお手軽なデータ通信が可能．
- 例えば，'a' という文字を送信する場合，'a' は 8'h61, 8'b01100001（次スライドのASCII Tableを参照）なので，下図のタイミングで送信線TXDを制御する．
  - データが送信されるまで送信線TXD を1とする．
  - まず，青色で示した0（これをスタートビットと呼ぶ）を送信することで，データ送信の開始を明示．
  - 次に，黄色で示した様に送信したいデータ 8'b01100001 の最下位ビットから順番に送信する．
  - 最後に，赤色で示した1（これをストップビットと呼ぶ）を送信する．
- 1ビットを送受信するための時間間隔は送信側と受信側で同じレートを用いる．これをボー・レート (baud) と呼ぶ．例えば，1000 baud であれば，1ビット送信の間隔は 1msec となる．



ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

シリアル通信による送信回路 m_UartTx

- システムクロック 50MHz, 1Mbaud を想定する送信回路
- トップのモジュール m_main では，2秒に1回の頻度で，文字 a を送信する.

code201.v

```
/*******************************************************************************/
/* code201.v                 For CSC.T363 Computer Architecture, Archlab TOKYO TECH */
/*******************************************************************************/
`timescale 1ns/100ps
`default_nettype none
/*******************************************************************************/
module m_main (w_clk100, w_txd);
   input  wire w_clk100;
   output wire w_txd;
   reg r_clk = 0;
   always@(posedge w_clk100) r_clk <= ~r_clk; // 50MHz clock signal
   reg [31:0]  r_cnt = 1;
   always@(posedge r_clk) r_cnt <= (r_cnt>(100_000_000 - 1)) ? 0 : r_cnt+1;
   m_UartTx m_UartTx0(r_clk, 8'h61, (r_cnt==0), w_txd);
endmodule
/*******************************************************************************/
module m_UartTx (w_clk, w_din, w_we, w_txd);
   input wire       w_clk, w_we;
   input wire [7:0] w_din;
   output wire      w_txd;
   reg [8:0]        r_buf = 9'b111111111;
   reg [7:0]        r_wait= 0;
   always@(posedge w_clk) begin
      r_wait <= (w_we) ? 0 : (r_wait>=49) ? 0 : r_wait + 1;
      r_buf  <= (w_we) ? {w_din, 1'b0} : (r_wait>=49) ? {1'b1, r_buf[8:1]} : r_buf;
   end
   assign w_txd = r_buf[0];
endmodule
/*******************************************************************************/
```