



**Dariusz Makowski**

**Katedra Mikroelektroniki i Technik  
Informatycznych**

**tel. 631 2720**

**[dmakow@dmcs.pl](mailto:dmakow@dmcs.pl)**

**<http://neo.dmcs.p.lodz.pl/sw>**



## Sprawy formalne

- ▶ Informacje wstępne
- ▶ Zaliczenie
- ▶ Laboratorium
- ▶ Materiały do wykładu



# Strona przedmiotu – materiały do wykładu

**DMCS Pages for Students - Mozilla Firefox**

File Edit View History Bookmarks Tools Help

http://heo.dmcs.p.lodz.pl/sw

C++ Conference Desy DMCS FPGA Mem MySQL 131.169.149.195 PCIe Perplexus RadMon RadMon2.5 RadMon (53)

AVG Search Active Surf-Shield Search-Shield AVG Info Get More

**Technical University of Lodz**  
**Department of Microelectronics and Computer Science**

W3C HTML 4.01

**Prezentacja specjalności prowadzonych przez DMCS**

**Strony domowe przedmiotów**

- Parallel and Distributed Programming
- Podstawy Energoelektroniki (EiT studia zaoczne sem. VIII)
- Podstawy Mikroelektroniki (Elektronika i Telekomunikacja sem. VI)
- Podstawy Programowania (EiT)
- Practical introduction to operating systems (IFE sem. II)
- Programming and Data Structures in C (IFE sem. II)
- Systemy wbudowane (Informatyka sem. IV)**
- Technologie handlu elektronicznego (Inf. sem IX)
- Technika kompilacji
- Technika Mikroprocesorowa (Informatyka sem. IV)

Find: mobil Next Previous Highlight all Match case

Done zotero



## Zakres przedmiotu

- ▶ Systemy mikroprocesorowe, systemy wbudowane
- ▶ Laboratorium
- ▶ Rodzina procesorów ARM
- ▶ Urządzenia peryferyjne
- ▶ Pamięci i dekodery adresowe
- ▶ Programy wbudowane na przykładzie procesorów ARM
- ▶ Metodyki projektowania systemów wbudowanych
- ▶ Interfejsy w systemach wbudowanych
- ▶ Systemy czasu rzeczywistego

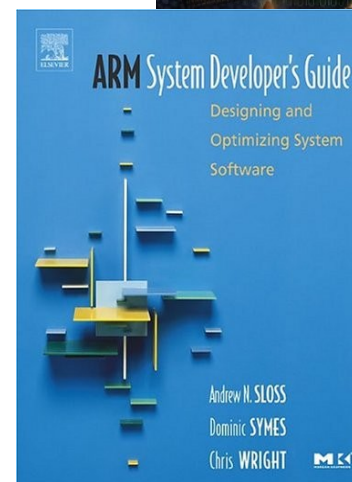
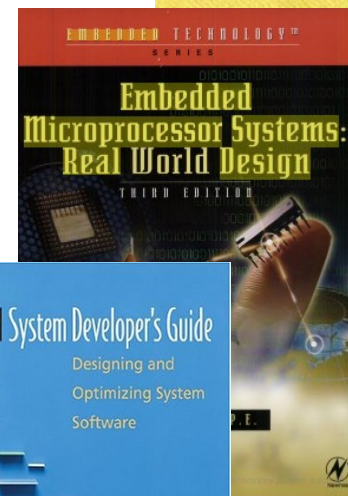
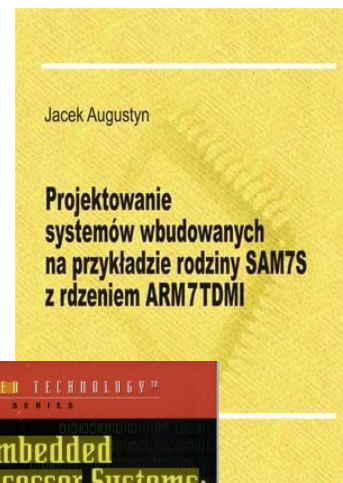


## Literatura obowiązkowa:

- Materiały wykładowe i laboratoryjne
- J. Augustyn, „Projektowanie systemów wbudowanych na przykładzie rodziny SAM7S z rdzeniem ARM7TDMI”, IGSMiE PAN, 2007, ISBN: 978-83-60195-55-0
- A. Sloss, D. Symes, C. Wright, „ARM System Developer's Guide: Designing and Optimizing System Software”, Elsevier, 2004
- S. R. Ball, “Embedded Microprocessor Systems: Real World Design”, Elsevier Science, 2002

## Literatura uzupełniająca:

- Andrew S. Tanenbaum „Strukturalna organizacja systemów komputerowych”, wydanie V, Helion 2006





## Zakres przedmiotu

- ▶ Systemy mikroprocesorowe, systemy wbudowane
- ▶ Laboratorium
- ▶ Rodzina procesorów ARM
- ▶ Urządzenia peryferyjne
- ▶ Pamięci i dekodery adresowe
- ▶ Programy wbudowane na przykładzie procesorów ARM
- ▶ Metodyki projektowania systemów wbudowanych
- ▶ Interfejsy w systemach wbudowanych
- ▶ Systemy czasu rzeczywistego



## Definicje podstawowe (1)

### ★ **Procesor (ang. Processor, Central Processing Unit)**

Urządzenie cyfrowe, sekwencyjne, potrafiące pobierać dane z pamięci, interpretować je i wykonywać jako rozkazy

### ★ **Mikroprocesor (ang. Microprocessor)**

Układ cyfrowy wykonany jako pojedynczy układ scalony o wielkim stopniu integracji (VLSI) zdolny do wykonywania operacji cyfrowych według dostarczonych mu informacji, np.: x86, Z80, 68k

### ★ **Mikrokontroler (ang. Microcontroller)**

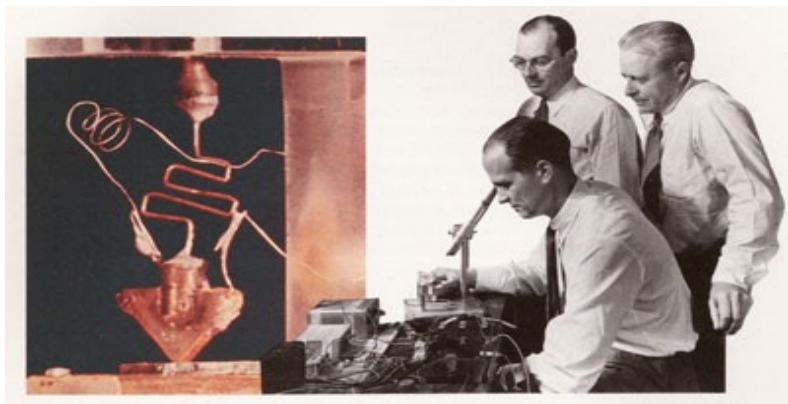
Komputer wykonany w jednym układzie scalonym, używany do sterowania urządzeniami elektronicznymi. Oprócz jednostki centralnej CPU posiada zintegrowane pamięci oraz urządzenia peryferyjne, np.: Intel 80C51, Atmel Atmega128, Freescale MCF5282, ARM926EJ-S



## Historia mikroprocesorów (1)

1940 – Russell Ohl – demonstracja złącza półprzewodnikowego (dioda germanowa, bateria słoneczna)

1947 – Shockley, Bardeen, Brattain prezentują pierwszy tranzystor



Pierwszy tranzystor, Bell Laboratories



Pierwszy układ scalony, TI

1958 – Jack Kilby wynalazł pierwszy układ scalony

1967 – Laboratorium Fairchild oferuje pierwszą pamięć nieulotną ROM (64 bity)

1969 – Noyce i Moore opuszczają laboratorium Fairchild, powstaje niewielka firma INTEL. INTEL produkuje pamięci SRAM (64 bity). Japońska firma Busicom zamawia 12 różnych układów realizujących funkcje kalkulatorów.

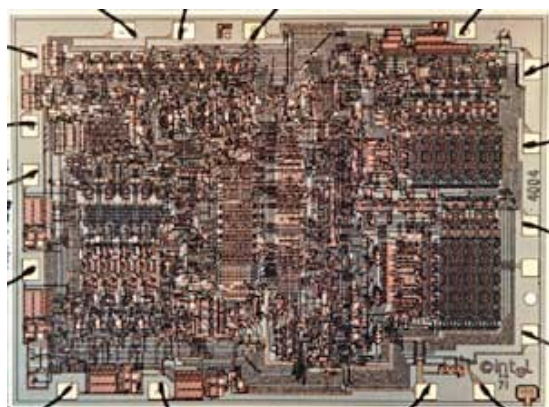




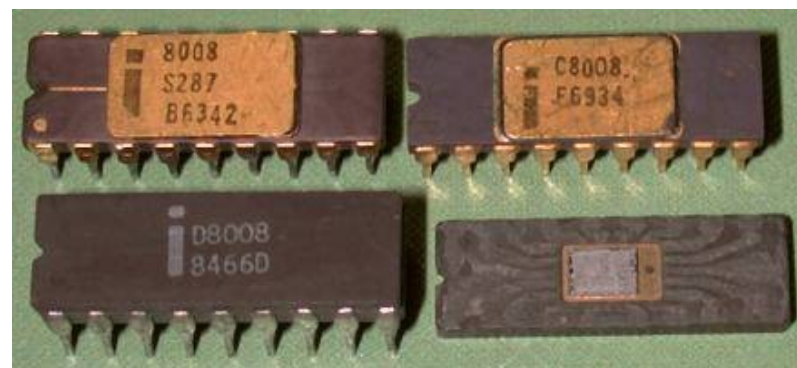
## Historia mikroprocesorów (2)

1970 - **F14 CADC** (Central Air Data Computer) mikroprocesor zaprojektowany przez Steve'a Gellera i Raya Holta na potrzeby armii amerykańskiej (myśliwiec F-14 Tomcat)

1971 - **Intel 4004** 4-bitowy procesor realizujące funkcje programowalnego kalkulatora (powszechnie uznaje się za pierwszy na świecie mikroprocesor), 3200 tranzystorów. INTEL wznawia pracę nad procesorami, Faggin z Fairchild pomaga rozwiązać problemy.



Zdjęcie 4-bitowego procesora INTEL 4004



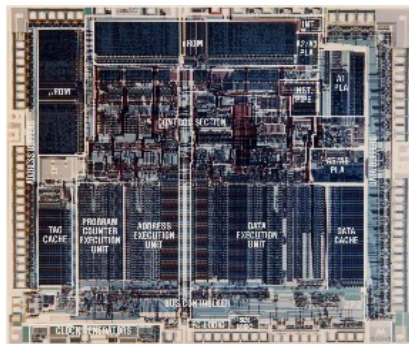
8-bitowe procesory INTEL-a

1972 – Faggin rozpoczyna prace nad 8-bitowym procesorem INTEL 8008. Rynek zaczyna się interesować układami “programowalnymi” - procesorami.

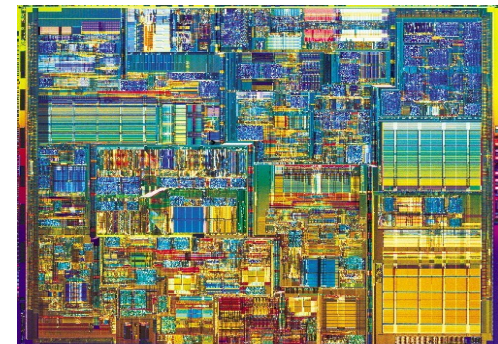


## Historia mikroprocesorów (3)

- 1974 – INTEL wprowadza na rynek ulepszona wersję 8008, procesor Intel 8080.  
Faggin opuszcza firmę Intel i zakłada firmę o nazwie Zilog. Motorola oferuje 8-bitowy procesor 6800 (NMOS, 5 V).
- 1975 – 8-bitowy procesor INTEL 6502 (technologia MOS) – najtańszy proc. na rynku.
- 1978 – Pierwszy 16-bitowy procesor 8086 (ulepszony 8080).
- 1979 – Motorola oferuje 16-bitowy procesor 68000.
- 1980 – Motorola wprowadza nowy 32-bitowy procesor 68020, 200,000 tranzystorów.



Motorola 68020



Intel, Pentium 4 Northwood

Intel 386, 486, Pentium I, II, III, IV, Centrino, Pentium D, Duo/Quad core, ...

Motorola 68030, 68040, 68060, PowerPC, ColdFire, ARM 7, ARM 9, StrongARM, ...



## Definicje podstawowe (2)

### ★ Komputer (ang. Computer)

Urządzenie elektroniczne, maszyna cyfrowa zdolna do przetwarzania danych cyfrowych zgodnie z dostarczonym programem

### ★ Komputer (system) wbudowany (ang. Embedded Computer)

Dedykowany system komputerowy, niewielkich rozmiarów sterownik wbudowany w urządzenie, przeznaczony do sterowania urządzeniem mechanicznym, elektrycznym lub elektronicznym

### ★ Komputer osobisty (ang. Personal Computer)

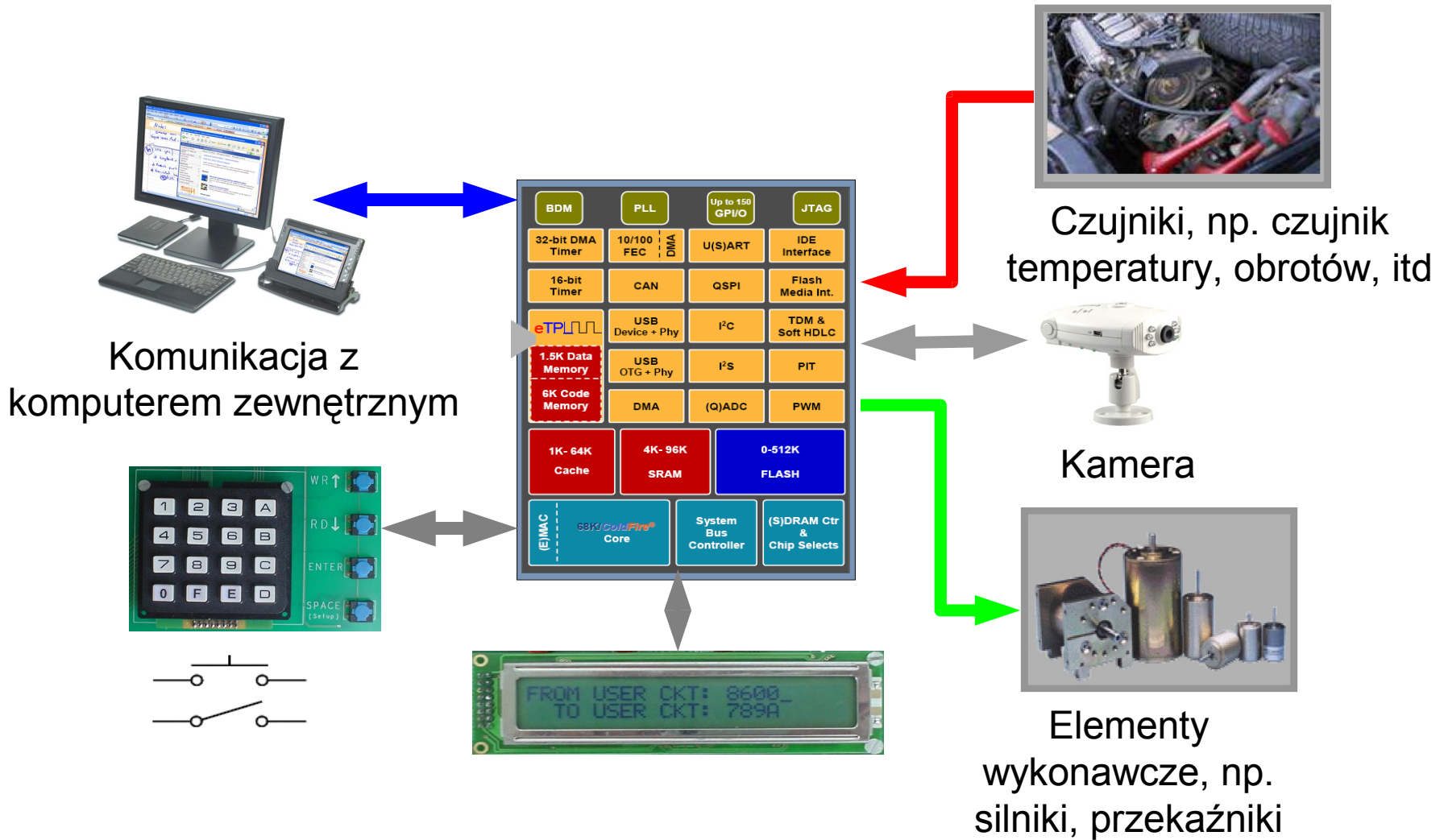
System komputery przeznaczony do użytku osobistego, domowego lub biurowego. Komputer wyposażony w system operacyjny przeznaczony do wykonywania aplikacji wykorzystywanych przez użytkownika

### ★ Architektura komputera (ang. Computer Architecture)

- Sposób organizacji oraz współpracy podstawowych elementów systemu komputerowego, tj. procesora, pamięci oraz urządzeń peryferyjnych
- Opis komputera z punktu widzenia programisty w języku niskiego poziomu (assembler). Budowa procesora, potoku wykonawczego oraz model programowy procesora

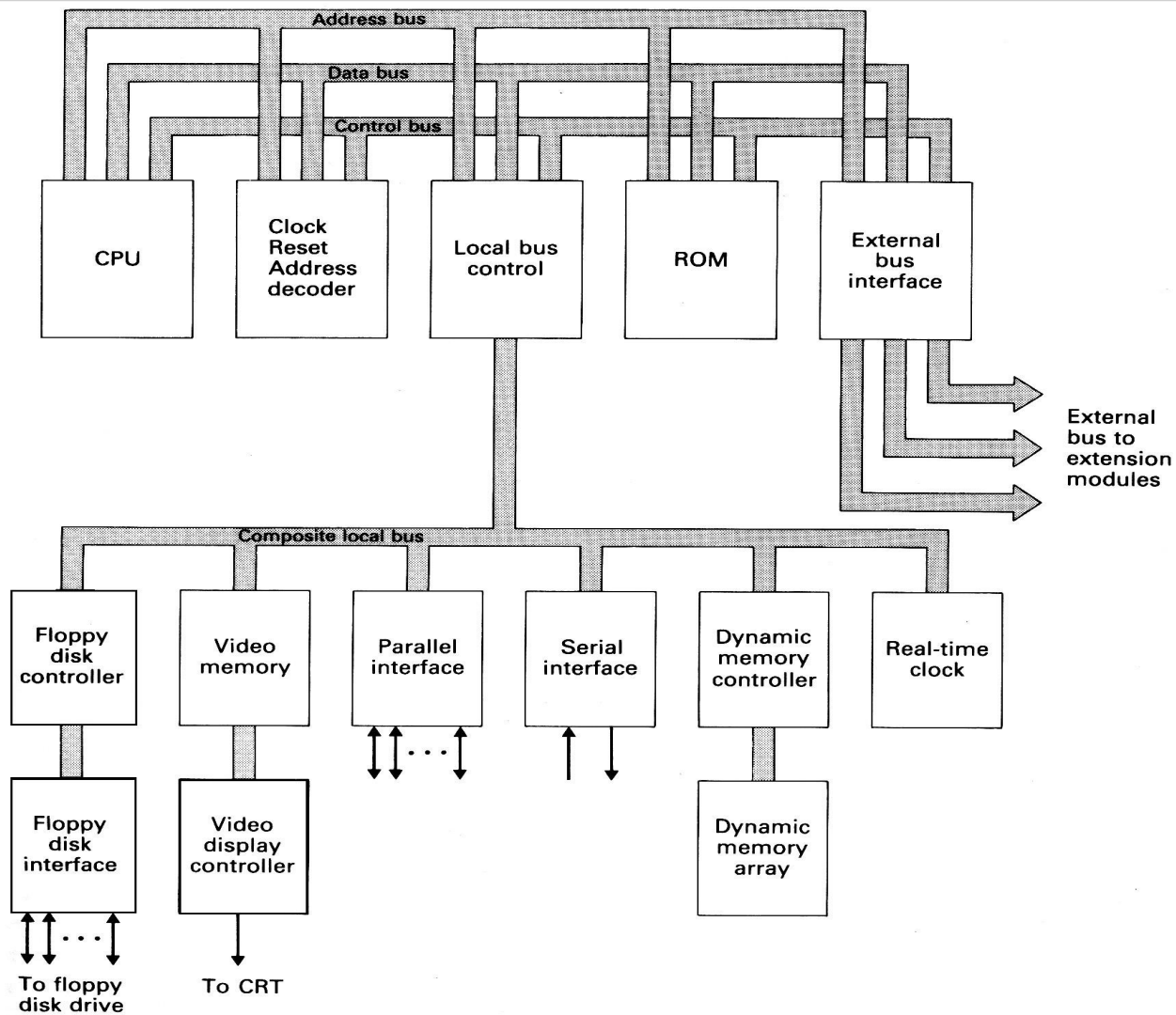


# Komputer wbudowany (embedded computer)



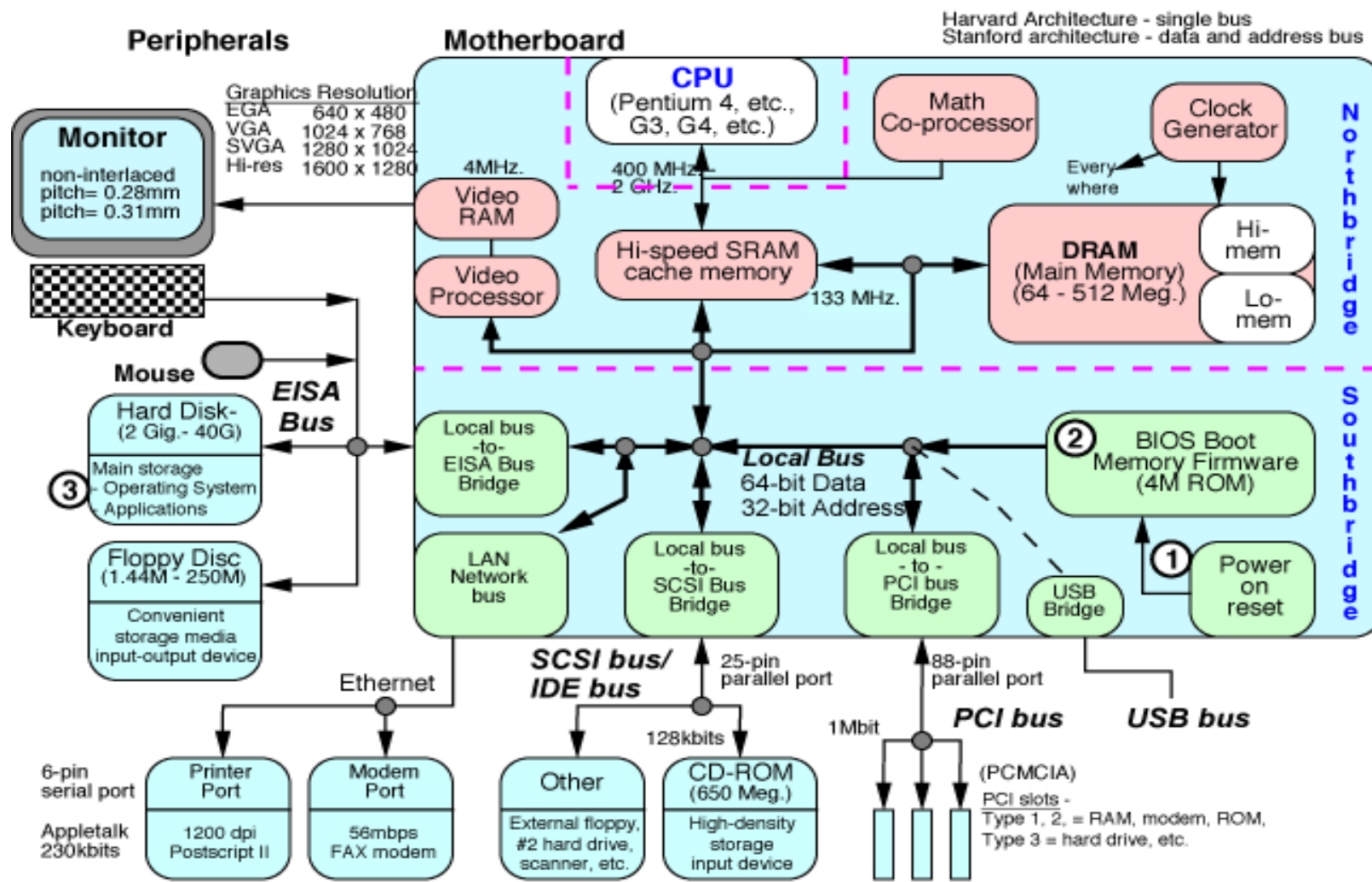


# Komputer uniwersalny





# Schemat blokowy komputera osobistego





## Definicje podstawowe (3)

### ★ **Pamięć komputerowa (ang. Computer Memory)**

Pamięć komputerowa to urządzenie elektroniczne lub mechaniczne służące do przechowywania danych i programów (systemu operacyjnego oraz aplikacji).

### ★ **Urządzenia zewnętrzne, peryferyjne (ang. Peripheral Device)**

Urządzenia elektroniczne dołączone do procesora przez magistrale systemową lub interfejs. Urządzenia zewnętrzne wykorzystywane są do realizowania specjalizowanej funkcjonalności systemu.

### ★ **Magistrala (ang. bus)**

Połączenie elektryczne umożliwiające przesyłanie danym pomiędzy procesorem, pamięcią i urządzeniami peryferyjnymi. Magistra systemowa zbudowane jest zwykle z kilkudziesięciu połączeń elektrycznych (ang. Parallel Bus) lub szeregowego połączenia (ang. Serial Bus).

### ★ **Interface (ang. Interface)**

Urządzenie elektroniczne lub optyczne pozwalające na komunikację między dwoma innymi urządzeniami, których bezpośrednio nie da się ze sobą połączyć.



## Definicje podstawowe (4)

### ★ Komputer SoC (ang. System-on-Chip)

Układ scalony wielkiej integracji zawierający **kompletny system elektroniczny zintegrowany z układami analogowymi, cyfrowo-analogowymi oraz radiowymi**. Poszczególne moduły tego systemu, ze względu na ich złożoność, pochodzą zwykle od różnych dostawców, np. rdzeń procesora od jednego producenta, układy peryferyjne od innego, interfejsy od jeszcze innego, itd...

Typowym obszarem zastosowań SoC są systemy wbudowane, a najbardziej rozpowszechnionym przedstawicielem tego rozwiązania są systemy oparte na procesorach ARM.

W przypadku, gdy nie jest możliwa integracja wszystkich układów na jednym podłożu półprzewodnikowym, poszczególne moduły wykonuje się na osobnych kryształach, a całość zamyka się w jednej obudowie, SiP (ang. System-in-a-package).

SoC różnią się od mikrokontrolerów **znacznie wydajniejszą jednostką obliczeniową CPU** (pozwalającej uruchamiać systemy operacyjne, np. Linux, Windows) oraz są zwykle **wyposażone w specjalizowane układy peryferyjne**.

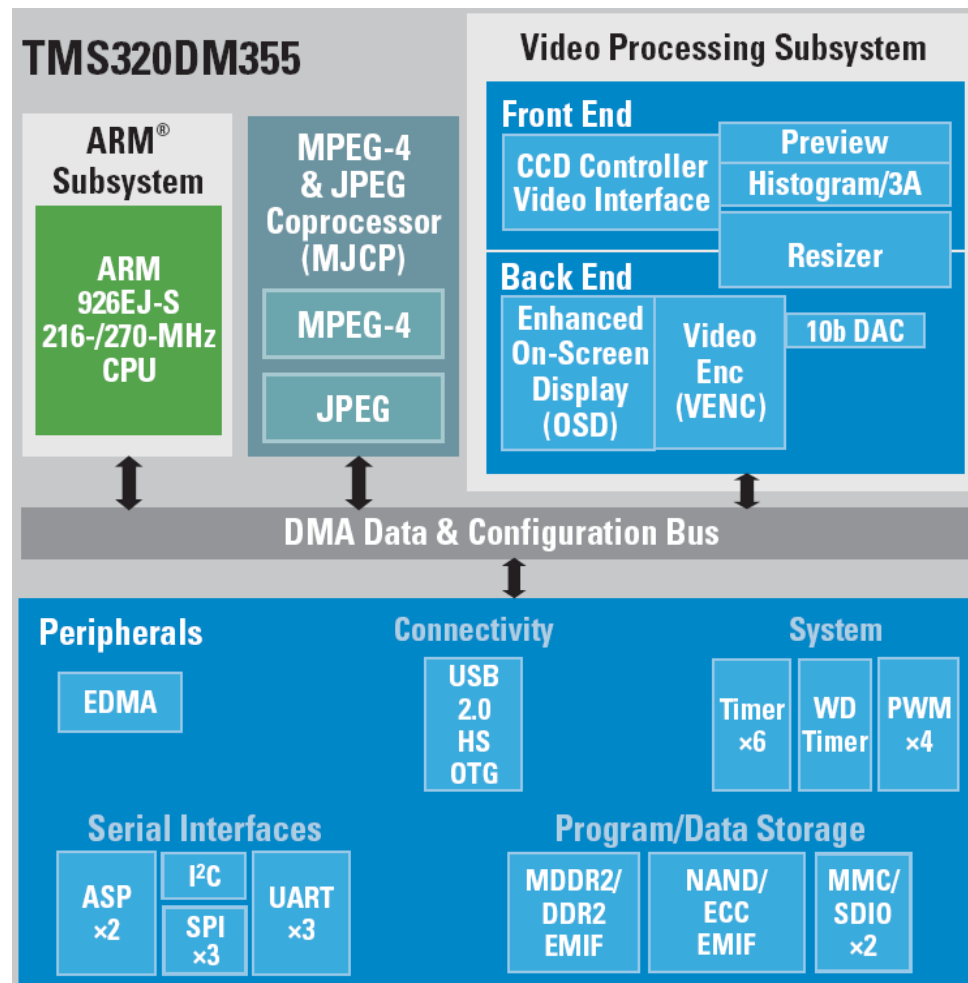




# SoC - DaVinci, digital media processor

## DaVinci DM355

- SoC opracowany przez firmę Texas Instruments
- Dedykowany co-procesor do przetwarzania dźwięku i obrazu w czasie rzeczywistym
- Niski pobór energii 400 mW podczas dekodowania HD MPEG4, 1 mW w stanie czuwania (systemy przenośne)
- Bogate interfejsy i układy peryferyjne (sterownik HDD, SD/MMC, USB, Ethernet,...)

Źródło: [www.ti.com](http://www.ti.com)



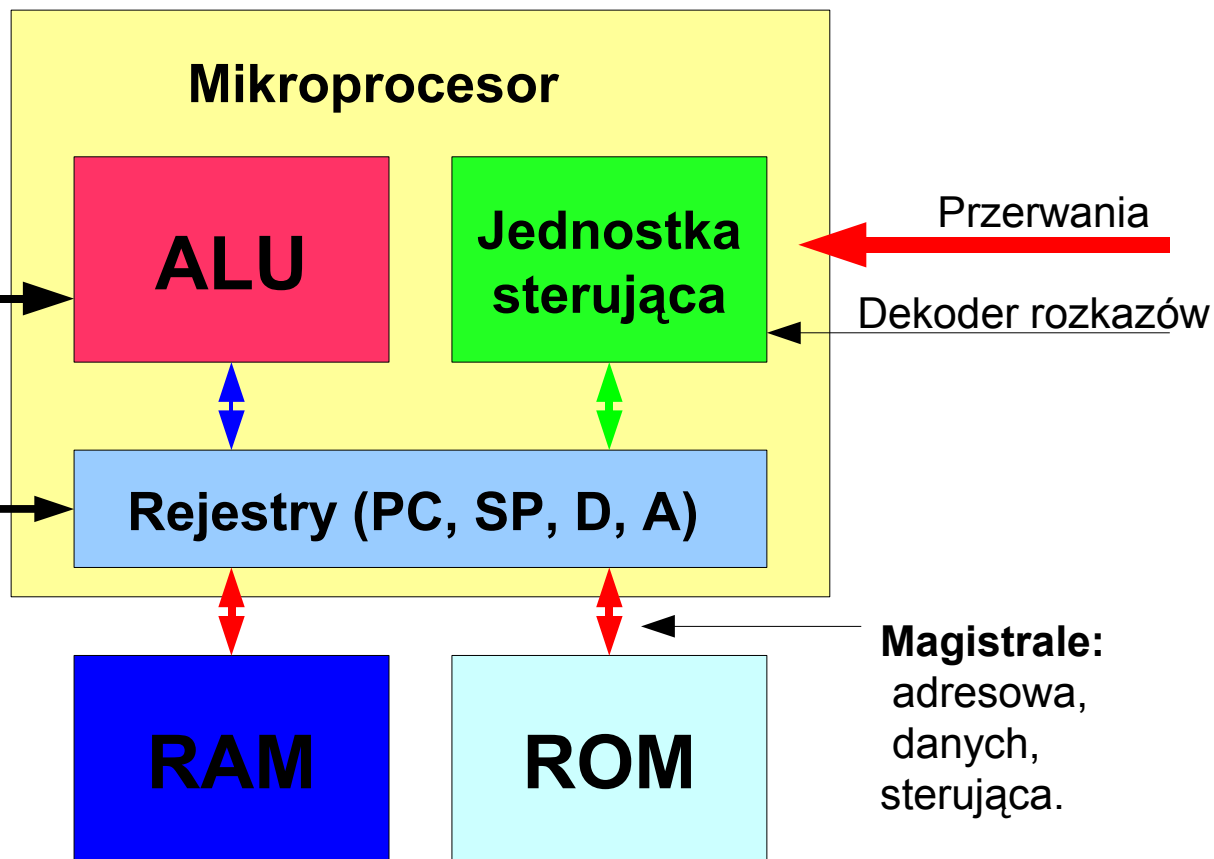
# Mikroprocesor

Mikroprocesor to układ cyfrowy wykonany jako pojedynczy układ scalony o wielkim stopniu integracji zdolny do wykonywania operacji cyfrowych według dostarczonych mu instrukcji.

Jednostka

arytmetyczno-logiczna  
(ang. Arithmetic Logic Unit),  
realizuje podstawowe  
operacje matematyczne  
8, 16, 32, 64-bit

Rejestry procesora -  
obszar (plik) rejestrów  
(ang. registers file)  
- komórki szybkiej pamięci  
statycznej, umieszczonej,  
wewnątrz procesora,  
8, 16, 32, 64, 128-bit,

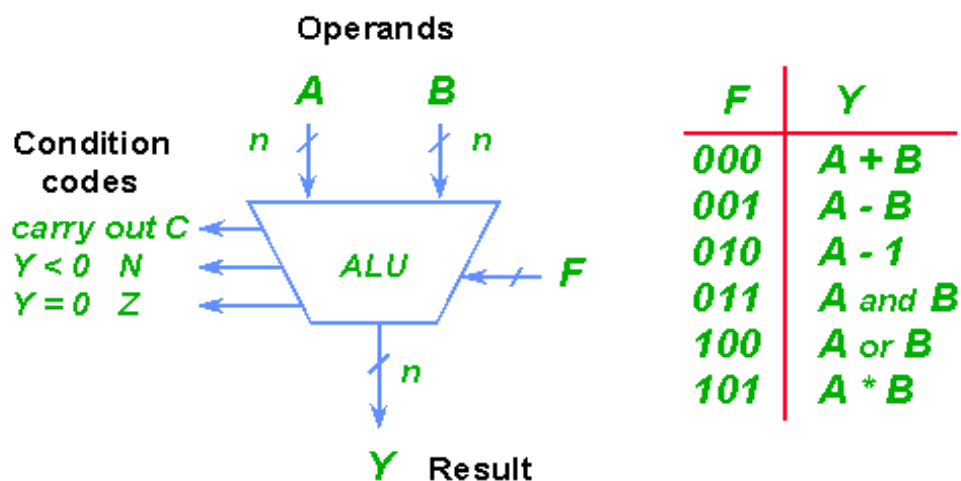




# Jednostka arytmetyczno-logiczna

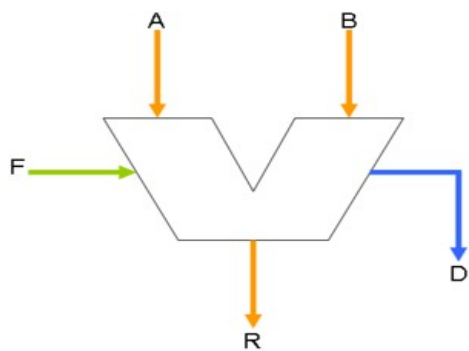
ALU wykorzystywana jest do wykonywania:

- operacji logicznych AND, OR, NOT, XOR,
- dodawania,
- odejmowania, negacja liczby, dodawanie z przeniesieniem,
- zwiększanie/zmniejszanie o 1,
- przesunięcia bitowe o stałą liczbę bitów,
- mnożenia i/lub dzielenia (dzielenie modulo).





# Dwubitowa jednostka ALU

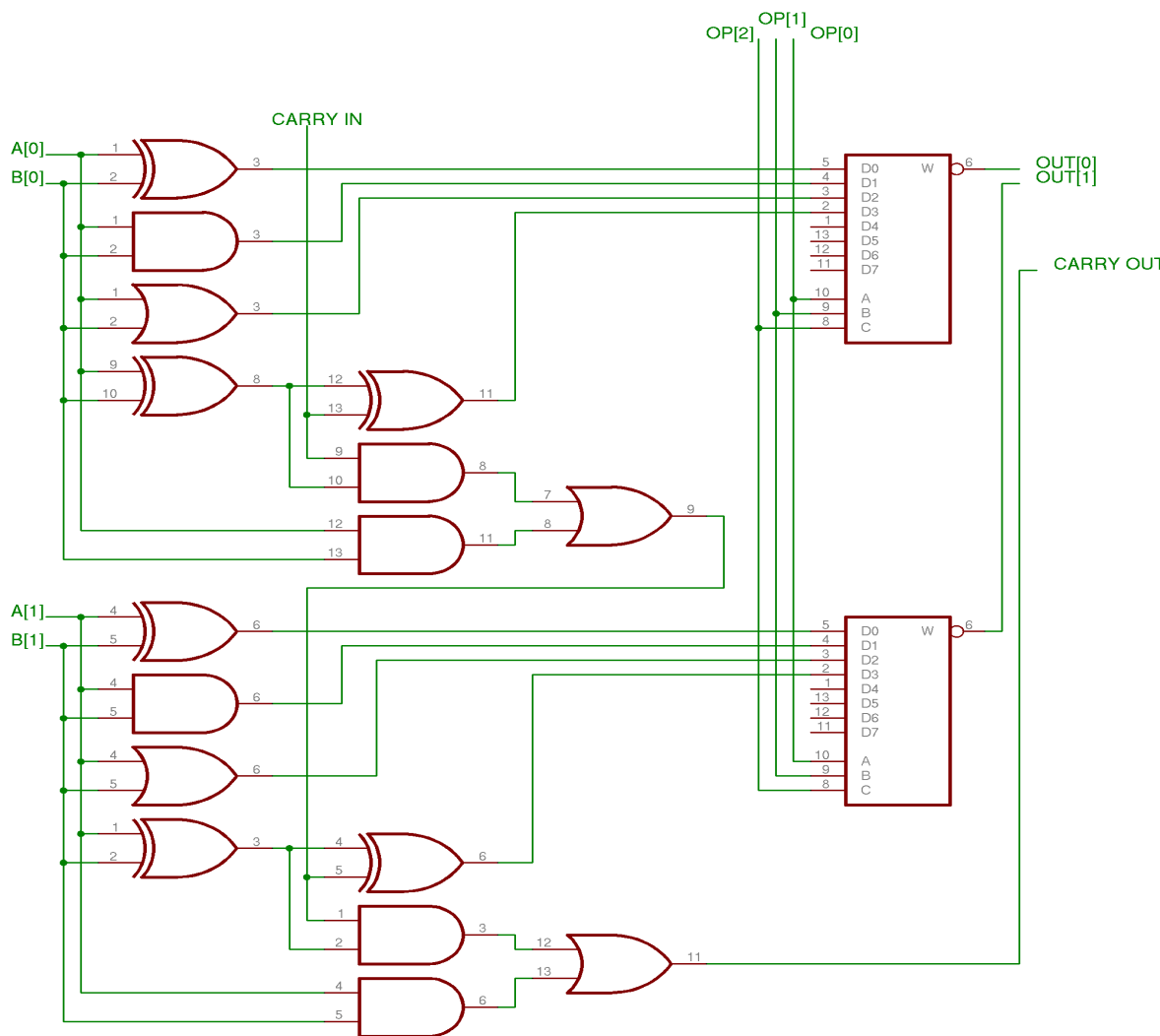


## Realizowane operacje:

- OP = 000 → XOR
- OP = 001 → AND
- OP = 010 → OR
- OP = 011 → ADD

## Inne możliwe operacje:

- subtraction,
- multiplication,
- division,
- NOT A,
- NOT B





## Architektura procesora (1)

Architektura procesora określa najważniejszych z punktu widzenia budowy i funkcjonalności cechy procesora.

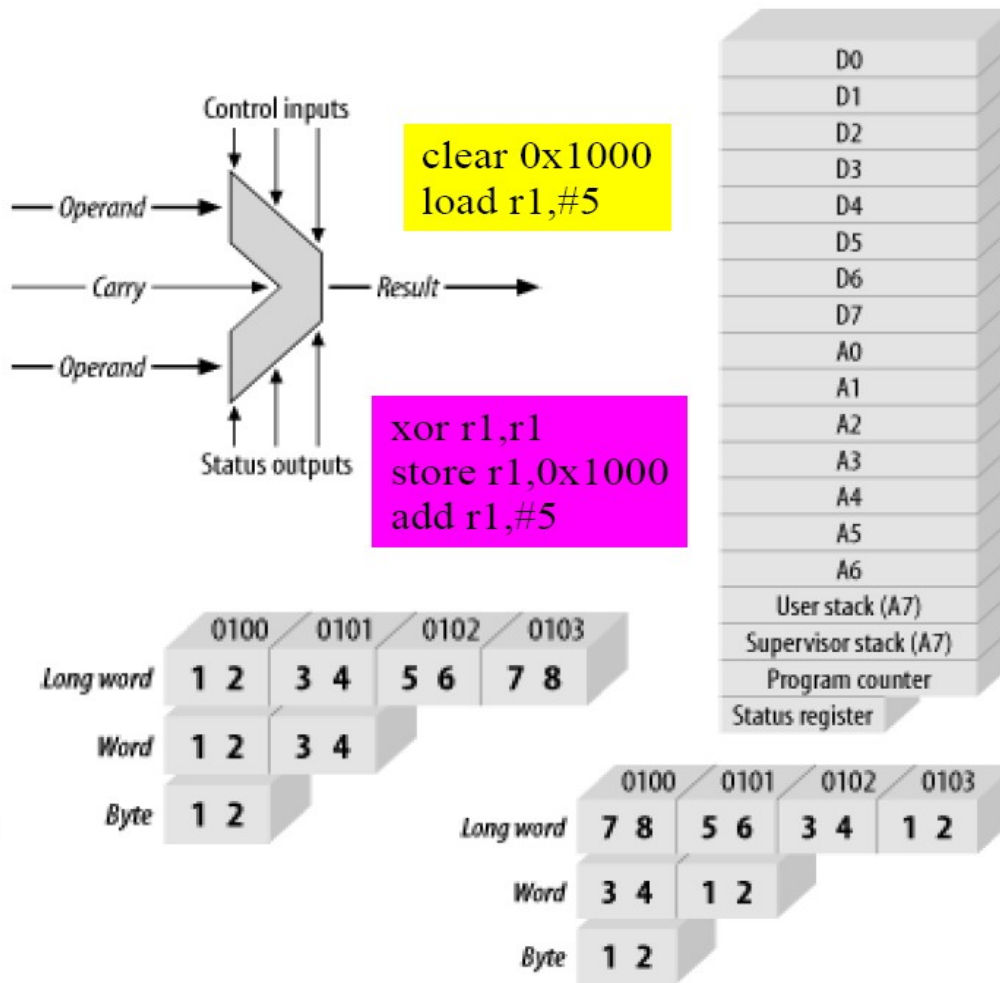
Na architekturę procesora składają się:

- model programowy procesora (ang. Instruction Set Architecture) – zestaw instrukcji procesora oraz inne jego cechy istotne z punktu widzenia programisty, bez względu na ich wewnętrzną realizację; stanowi granicę pomiędzy warstwą sprzętową a programową
- mikroarchitektura procesora (ang. microarchitecture) – wewnętrzna, sprzętowa implementacja danego modelu programowego, określająca sposób wykonywania operacji przez procesor, szczegółową budowę wewnętrzną procesora, itd.



# Architektura procesora (2)

- ALU
  - lista operacji
- zestaw rejestrów
  - A, I, PC, SR, SP,...
- lista rozkazów
  - CISC, RISC
- tryby adresowania
  - R, A, I,...
- przerwania
  - hardware, software
- big/little endian





# Architektura procesora CISC

## Cechy architektury CISC (Complex Instruction Set Computers):

- ★ Duża liczba rozkazów (instrukcji),
- ★ Niektóre rozkazy potrzebują dużej liczby cykli procesora do wykonania,
- ★ Występowanie złożonych, specjalistycznych rozkazów,
- ★ Duża liczba trybów adresowania,
- ★ Do pamięci może się odwoływać bezpośrednio duża liczba rozkazów,
- ★ Mniejsza od układów RISC częstotliwość taktowania procesora,
- ★ Powolne działanie dekodera rozkazów, ze względu na dużą ich liczbę i skomplikowane adresowanie

RISC / CISC



## Przykłady rodzin procesorów o architekturze CISC to:

- x86
- **M68000**
- PDP-11
- AMD



# Architektura procesora RISC

## Cechy architektury RISC (Reduced Instruction Set Computer):

- ★ Zredukowana liczba rozkazów. Upraszcza to znacznie dekodery rozkazów.
- ★ Redukcja trybów adresowania, dzięki czemu kody rozkazów są prostsze,
- ★ Ograniczenie komunikacji pomiędzy pamięcią, a procesorem. Do przesyłania danych pomiędzy pamięcią, a rejestrami służą dedykowane instrukcje (load, store).
- ★ Zwiększenie liczby rejestrów (np. 32, 192, 256),
- ★ Dzięki przetwarzaniu potokowemu (ang. pipelining) wszystkie rozkazy wykonują się w jednym cyklu maszynowym.

## Przykłady rodzin mikroprocesorów o architekturze RISC:

- ➔ IBM 801
- ➔ PowerPC
- ➔ MIPS
- ➔ Alpha
- ➔ **ARM**
- ➔ Motorola 88000
- ➔ ColdFire
- ➔ SPARC
- ➔ PA-RISC
- ➔ Atmel AVR

Obecnie produkowane procesory Intela z punktu widzenia programisty są widziane jako CISC, ale ich rdzeń jest zgodny z RISC. Rozkazy CISC są rozbijane na mikrorozkazy (ang. microops), które są następnie wykonywane przez szybki blok wykonawczy zgodny z architekturą RISC.

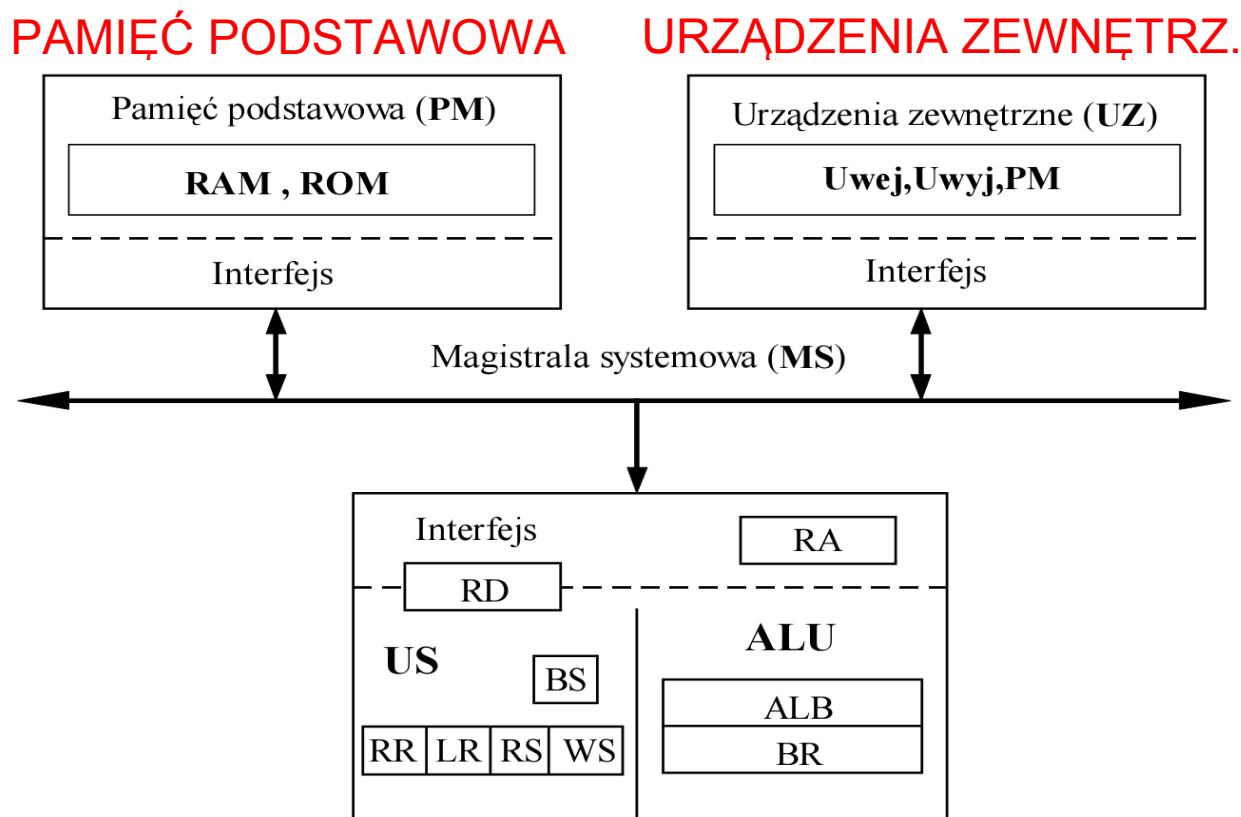




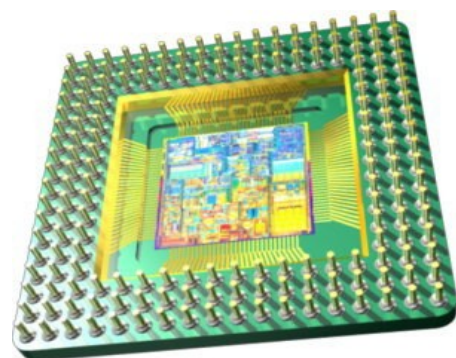
# Architektura systemu komputerowego

Architektura polega na ścisłym podziale komputera na trzy podstawowe części:

- ➔ procesor,
- ➔ pamięć (zawierająca dane oraz program),
- ➔ urządzenia wejścia/wyjścia (I/O).



# Magistrale komputera



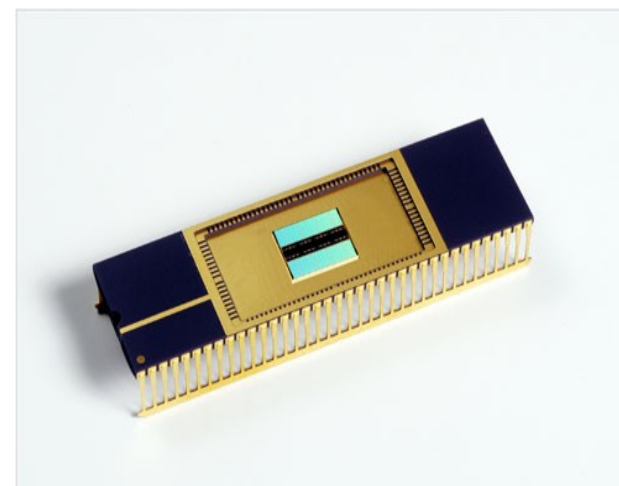
Magistrala adresowa



Magistrala danych



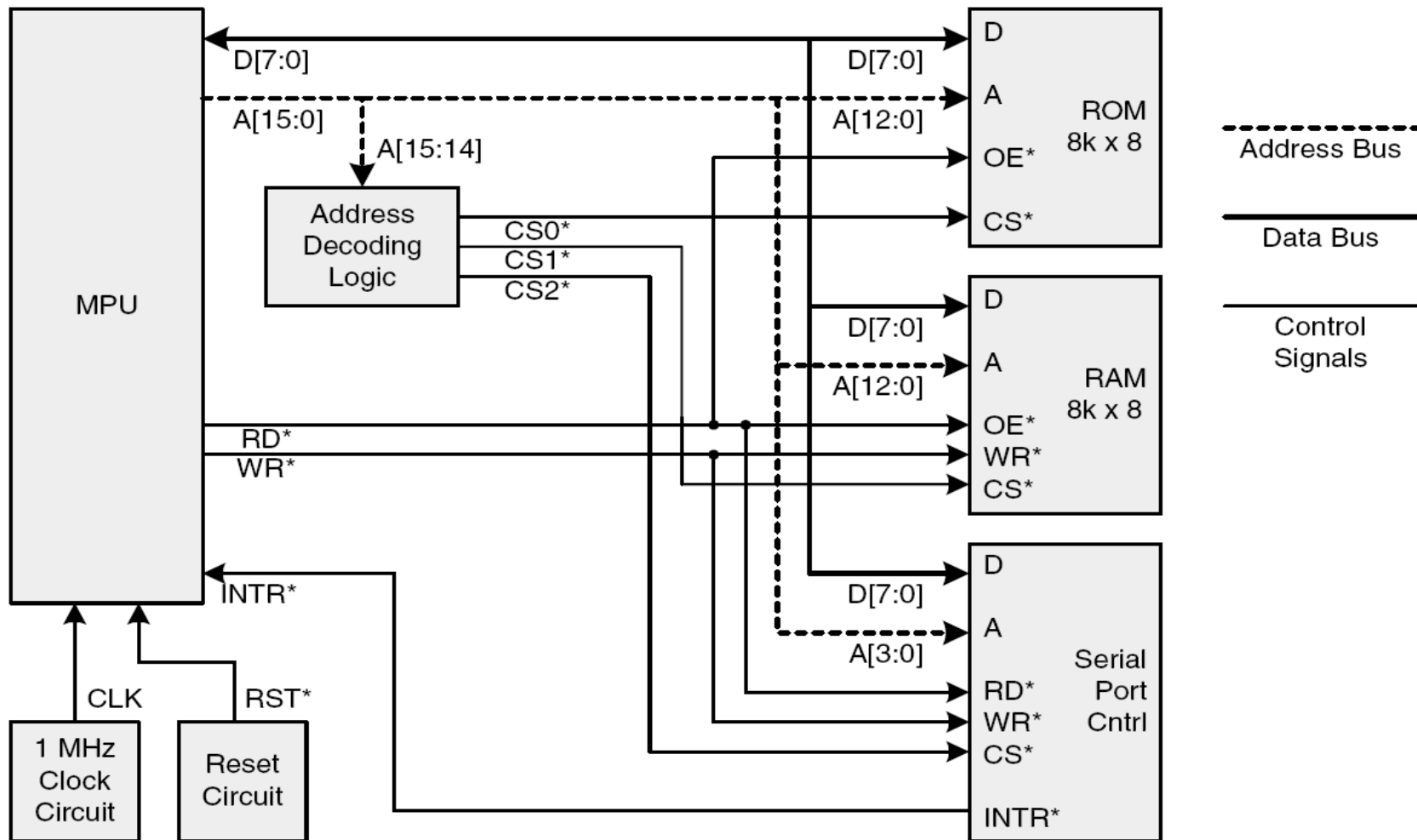
Magistrala sterująca



1. Rodzaj magistrali
2. Szerokość magistrali
3. Częstotliwość zegara – szybkość transmisji



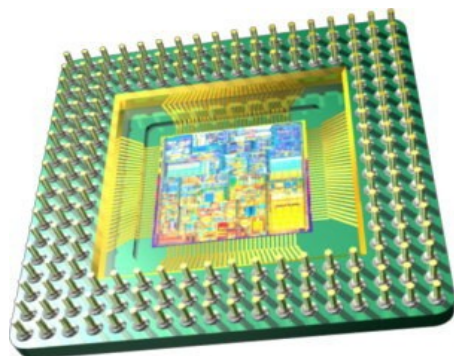
# Przykładowy komputer 8-bitowy



## Architektura von Neumanna

### Cechy architektury von Neumanna:

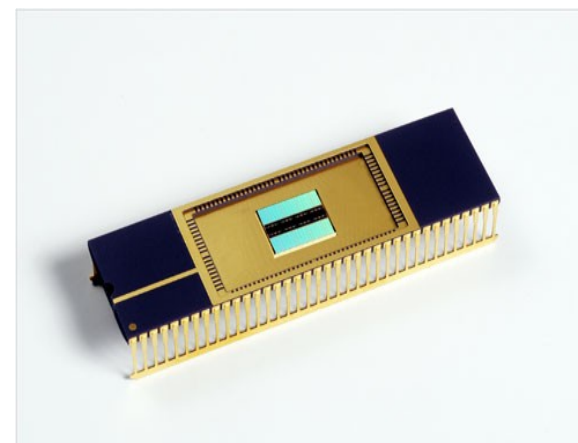
- ★ rozkazy i dane przechowywane są w tej samej pamięci,
- ★ nie da się rozróżnić danych o rozkazów (instrukcji),
- ★ dane nie mają przypisanego znaczenia,
- ★ pamięć traktowana jest jako liniowa tablica komórek, które identyfikowane są przy pomocy dostarczanego przez procesor adresu,
- ★ procesor ma dostęp do przestrzeni adresowej, dekodery adresowe zapewniają mapowanie pamięci na rzeczywiste układy.



Magistrala adresowa



Magistrala danych



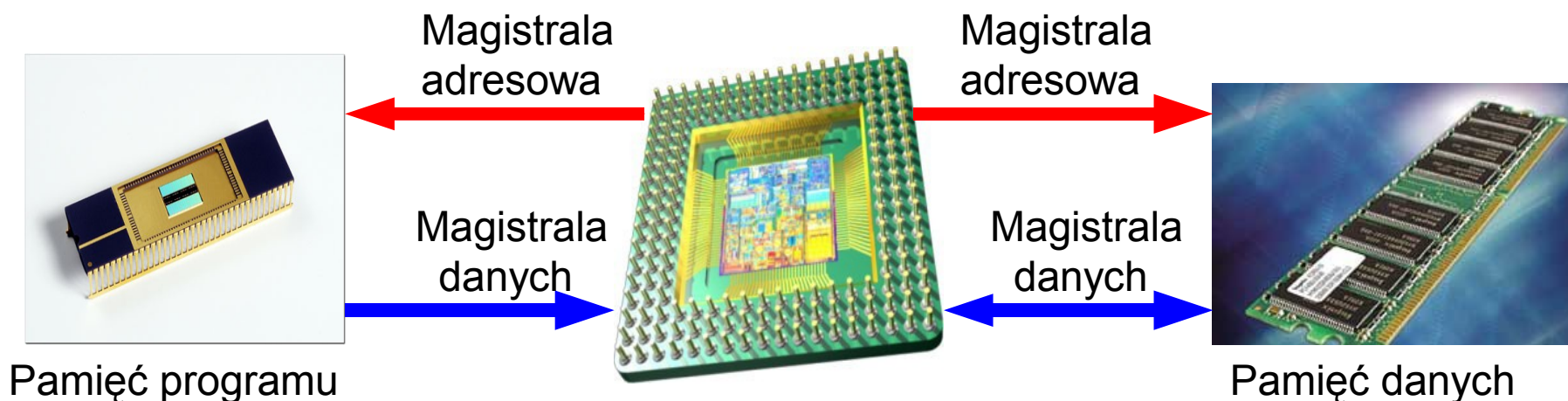


## Architektura Harwardzka

Prostsza (w stosunku do architektury Von Neumanna) budowa przekłada się na większą szybkość działania - dlatego ten typ architektury jest często wykorzystywany w procesorach sygnałowych oraz przy dostępie procesora do pamięci cache.

### Cechy architektury Harwardzkiej:

- ★ rozkazy i dane przechowywane są w oddzielnych pamięciach,
- ★ organizacja pamięci może być różna (inne długości słowa danych i rozkazów),
- ★ możliwość pracy równoległej – jednoczesny odczyt danych z pamięci programu oraz danych,
- ★ stosowana w mikrokontrolerach jednokładowych.

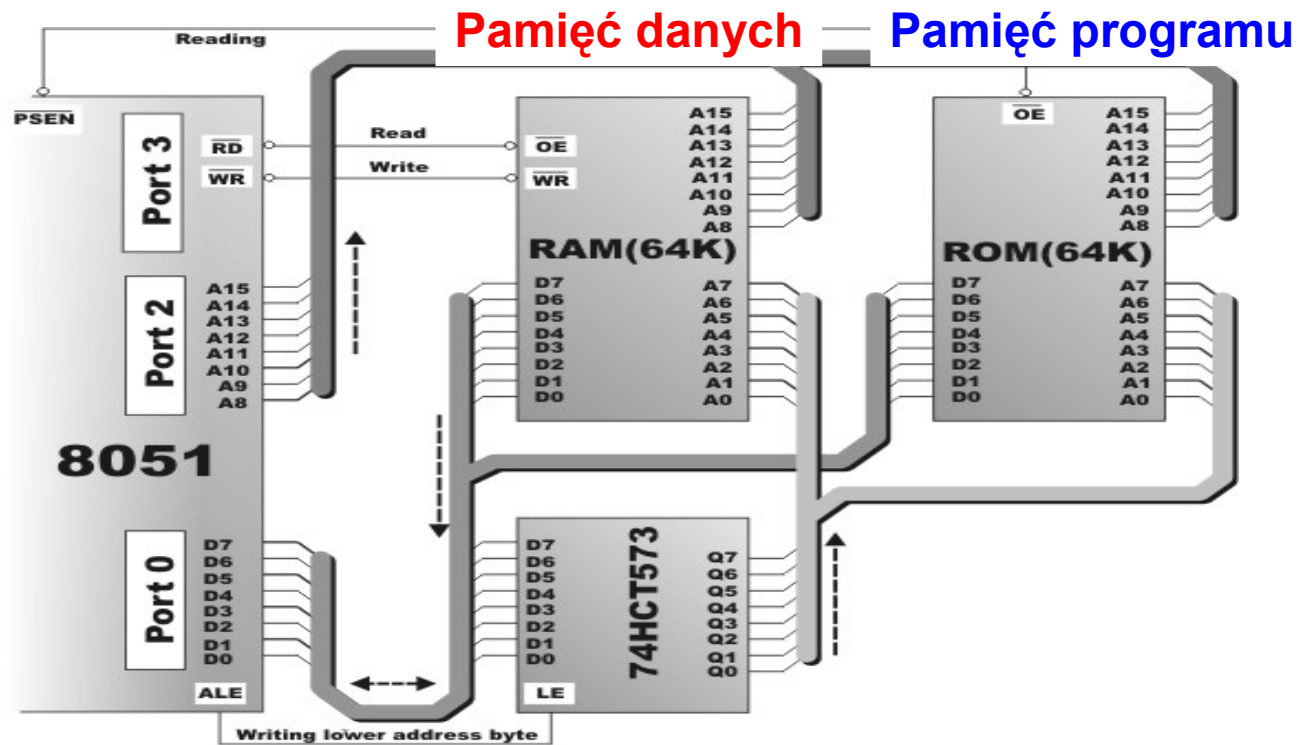




## Zmodyfikowana architektura harwardzka

### Zmodyfikowana architektura harwardzka (architektura mieszana)

- łączy w sobie cechy architektury harwardzkiej i architektury von Neumanna. Oddzielone zostały pamięci danych i rozkazów, lecz wykorzystują one wspólne magistrale danych i adresową. Architektura umożliwia łatwe przesyłanie danych pomiędzy rozdzielonymi pamięciami.



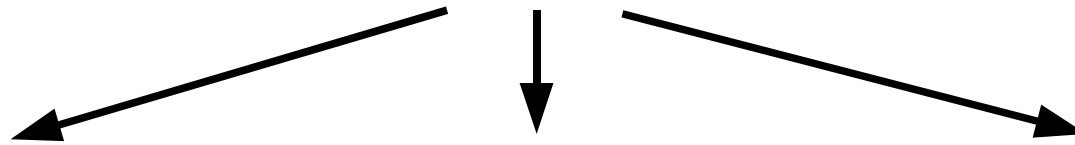
Przykład mikrokontrolera z rodziny 8051 wraz z zewnętrznymi pamięciami



## Kolejność bajtów w pamięci (1)

**Bajt** – najmniejsza adresowalna jednostka pamięci komputerowej

### Endianess



Big-endian

...pod najmłodszym adresem umieszczony jest najstarszy bajt

podobnie jak w języku polskim, angielskim

Motorola, SPARC, ARM

middle-endian

liczby zmiennoprzecinkowe podwójnej precyzji

VAX and ARM

Little-endian

...pod najstarszym adresem umieszczony jest najstarszy bajt

podobnie jak w językach arabskich, hebrajski

Intel x86, 6502 VAX

### Bi-Endian

ARM, PowerPC (za wyjątkiem PPC970/G5), DEC Alpha, MIPS, PA-RISC oraz IA64



## Kolejność bajtów w pamięci (2)

## Architektura 8-bitowa

	7	0
0x0000.0000		Byte 1
0x0000.0001		Byte 2
0x0000.0002		Byte 3
0x0000.0003		Byte 4
0x0000.0004		Byte 5

	7	0
0x0000.0000		0x12
0x0000.0001		0x34
0x0000.0002		0x56
0x0000.0003		0x78
0x0000.0004		0x90

Podwójne  
słowo (DW):  
**0x1234.5678**





## Kolejność bajtów w pamięci (3)

## Big-endian

Byte 4 ... Byte 1  
MSB                  LSB

0x0000.0000	Byte 4	Byte 3	Byte 2	Byte 1
0x0000.0004	Byte 8	Byte 7	Byte 6	Byte 5
0x0000.0008	Byte 12	...	...	...
0x0000.000C				
0x0000.0010				

## Little-endian

0x0000.0000	Byte 1	Byte 2	Byte 3	Byte 4
0x0000.0004	Byte 5	Byte 6	Byte 7	Byte 8
0x0000.0008	Byte 9	...	...	...
0x0000.000C				
0x0000.0010				



## Kolejność bajtów w pamięci (4)

Podwójne słowo (DW): **0x1234.5678**

### Big-endian

	32	24	23	16	15	8	7	0
0x0000.0000	<b>0x12</b>	<b>0x34</b>	<b>0x56</b>	<b>0x78</b>				
0x0000.0004	Byte 5	Byte 6	Byte 7	Byte 8				
0x0000.0008	Byte 9	...	...	...				
0x0000.000C								
0x0000.0010								

### Little-endian

	32	24	23	16	15	8	7	0
0x0000.0000	<b>0x78</b>	<b>0x56</b>	<b>0x34</b>	<b>0x12</b>				
0x0000.0004	Byte 8	Byte 7	Byte 6	Byte 5				
0x0000.0008	Byte 12	...	...	...				
0x0000.000C								
0x0000.0010								



## Kolejność bajtów w pamięci (5)

# Jak rozpoznać architekturę procesora oraz rozkład bajtów w pamięci?

```
#define LITTLE_ENDIAN 0
#define BIG_ENDIAN 1

int machineEndianness()
{
    long int i = 1;                /* 32 bit = 0x0000.0001 */
    const char *p = (const char *) &i; /* wskaźnik do .....? */

    if (p[0] == 1) /* Lowest address contains the least significant byte */
        return LITTLE_ENDIAN;

    else
        return BIG_ENDIAN;
}
```



## Kolejność bajtów w pamięci (5)

# Jak rozpoznać architekturę procesora oraz rozkład bajtów w pamięci?

```
#define LITTLE_ENDIAN 0
#define BIG_ENDIAN 1

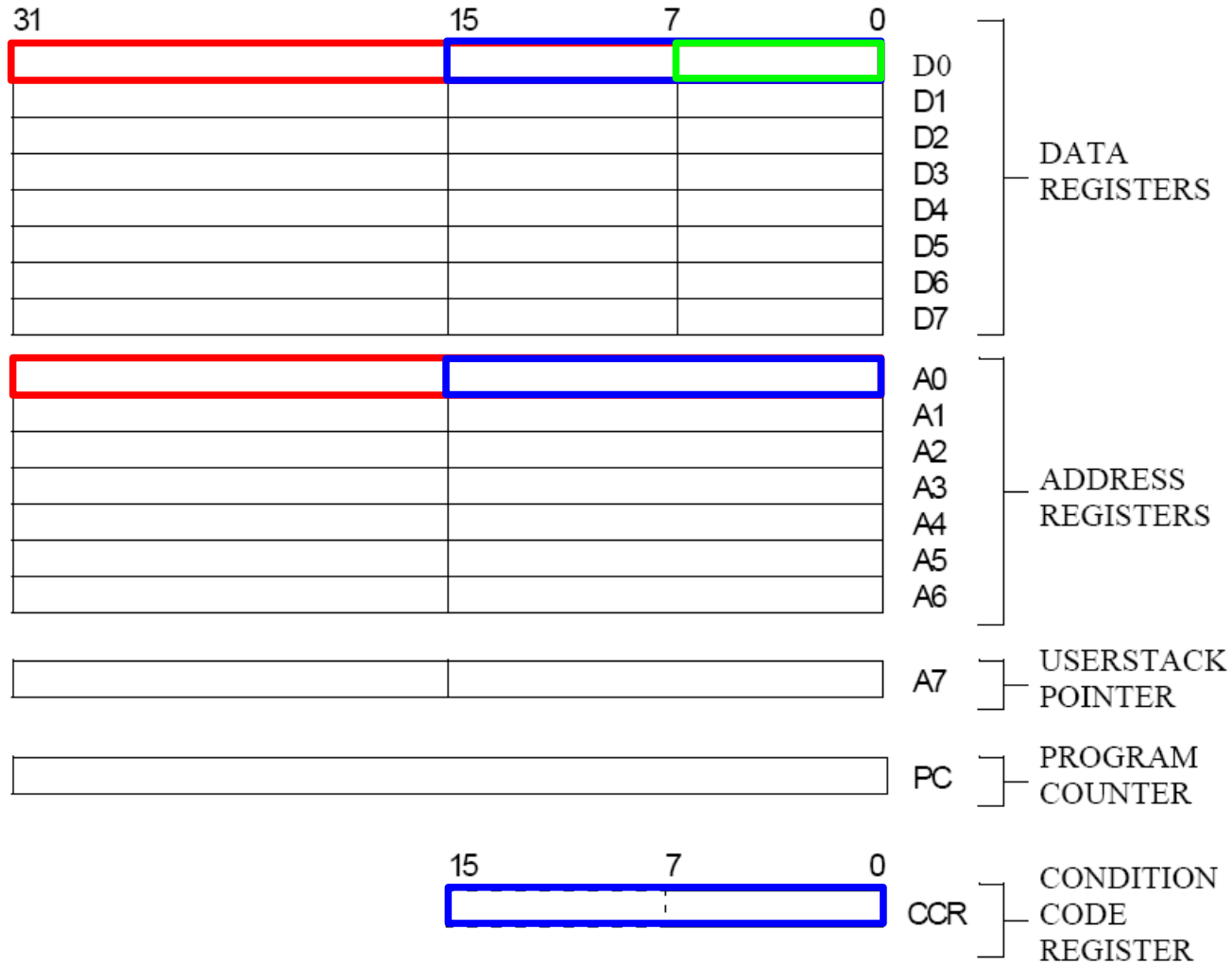
int machineEndianness()
{
    long int i = 1;                /* 32 bit = 0x0000.0001 */
    const char *p = (const char *) &i; /* wskaźnik do .....? */

    if (p[0] == 1) /* Lowest address contains the least significant byte */
        return LITTLE_ENDIAN;

    else
        return BIG_ENDIAN;
}
```



# Model programowy procesora ColdFire





## Przykład użycia rejestrów danych

$$y = \text{wsp. temp.} * \text{ADC} + \text{wsp. skalujacy}$$

ACC = wsp. temp.

ACC = ACC \* ADC

ACC = ACC + wsp. skalujacy

y = ACC

D0 = wsp. temp.

D1 = wsp. skalujacy

D2 = ADC

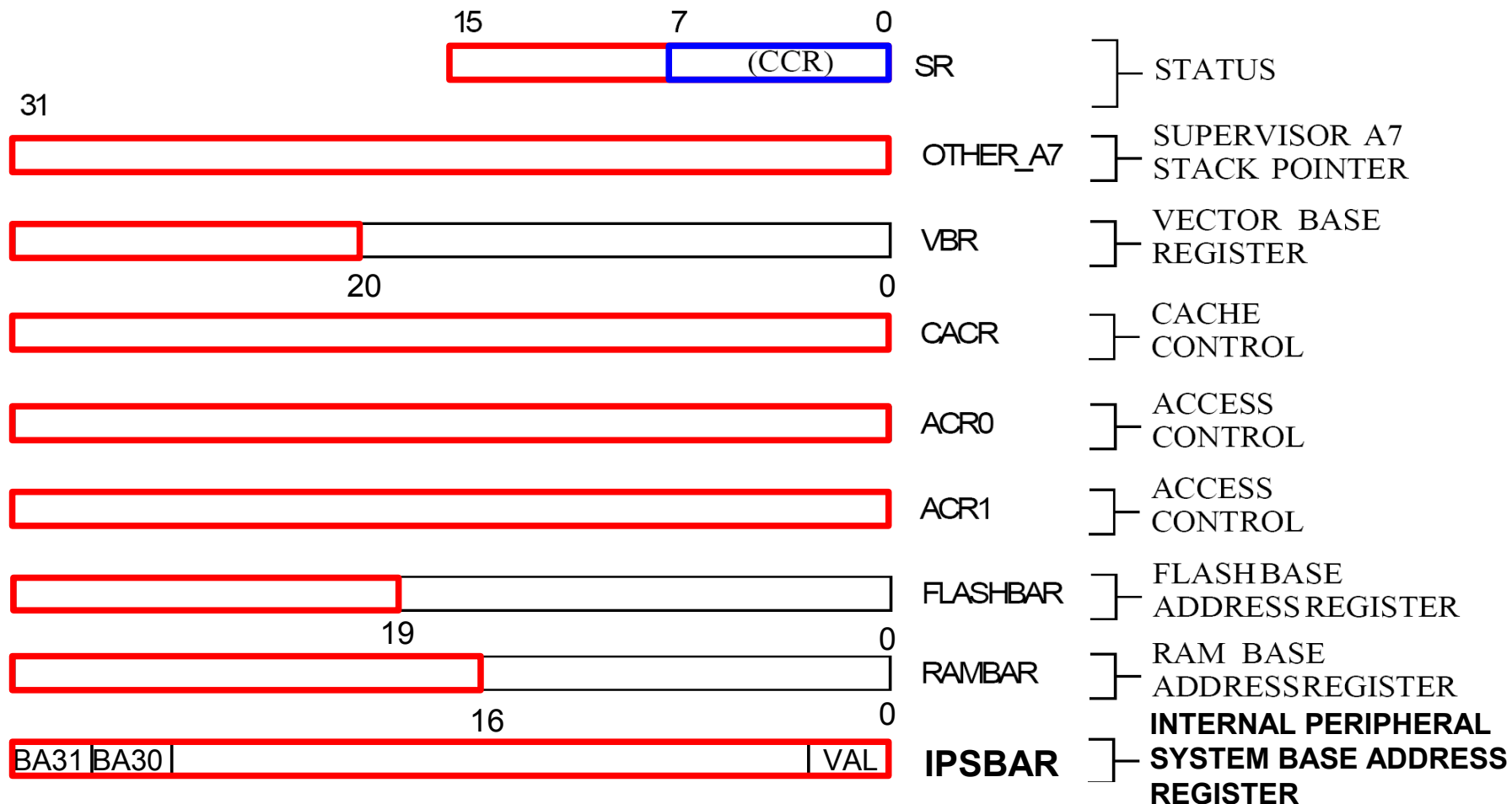
D2 = D0 \* D2

y = D2 + D1



# Model programowy procesora ColdFire

## Rejestry dostępne w trybie superużytkownika

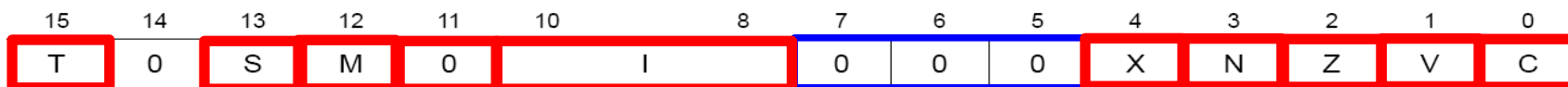




## Rejestry mikrokontrolera MCF5282

System Byte

Condition Code Register (CCR)



Bits	Name	Description
15	T	Trace enable. When set, the processor performs a trace exception after every instruction.
13	S	Supervisor/user state. Denotes whether the processor is in supervisor mode (S = 1) or user mode (S = 0).
12	M	Master/interrupt state. This bit is cleared by an interrupt exception, and can be set by software during execution of the RTE or move to SR instructions.
10–8	I	Interrupt level mask. Defines the current interrupt level. Interrupt requests are inhibited for all priority levels less than or equal to the current level, except the edge-sensitive level 7 request, which cannot be masked.
4	X	Extend condition code bit.
3	N	Negative condition code bit. Set if the most significant bit of the result is set; otherwise cleared.
2	Z	Zero condition code bit. Set if the result equals zero; otherwise cleared.
1	V	Overflow condition code bit. Set if an arithmetic overflow occurs implying that result cannot be represented in the operand size; otherwise cleared.
0	C	Carry condition code bit. Set if a carry out of the operand msb occurs for an addition, or if a borrow occurs in a subtraction; otherwise cleared Set to the value of the C bit for arithmetic operations; otherwise not affected.





## ADD

### Add

First appeared in ISA\_A

## ADD

**Operation:** Source + Destination → Destination

**Assembler Syntax:** ADD.L <ea>y,Dx  
ADD.L Dy,<ea>x

**Attributes:** Size = longword

**Description:** Adds the source operand to the destination operand using binary addition and stores the result in the destination location. The size of the operation may only be specified as a longword. The mode of the instruction indicates which operand is the source and which is the destination as well as the operand size.

The Dx mode is used when the destination is a data register; the destination <ea>x mode is invalid for a data register.

In addition, ADDA is used when the destination is an address register. ADDI and ADDQ are used when the source is immediate data.

**Condition Codes:**

X	N	Z	V	C
*	*	*	*	*

- X Set the same as the carry bit
- N Set if the result is negative; cleared otherwise
- Z Set if the result is zero; cleared otherwise
- V Set if an overflow is generated; cleared otherwise
- C Set if an carry is generated; cleared otherwise



# Instrukcja porównująca argumenty

## CMP

### Compare

First appeared in ISA\_A  
.B and .W First appeared in ISA\_B

## CMP

**Operation:** Destination – Source → cc

**Assembler Syntax:** CMP.sz <ea>y,Dx

**Attributes:** Size = byte, word, longword (byte, word supported starting with ISA\_B)

**Description:** Subtracts the source operand from the destination operand in the data register and sets condition codes according to the result; the data register is unchanged. The operation size may be a byte, word, or longword. CMPA is used when the destination is an address register; CMPI is used when the source is immediate data.

**Condition Codes:**

X	N	Z	V	C
—	*	*	*	*

X Not affected  
N Set if the result is negative; cleared otherwise  
Z Set if the result is zero; cleared otherwise  
V Set if an overflow occurs; cleared otherwise  
C Set if a borrow occurs; cleared otherwise



# Instrukcja skoku warunkowego

## Bcc

### Branch Conditionally

First appeared in ISA\_A  
.L First appeared in ISA\_B

## Bcc

**Operation:** If Condition True  
Then  $PC + d_n \rightarrow PC$

**Assembler Syntax:** Bcc.sz <label>

**Attributes:** Size = byte, word, longword (longword supported starting with ISA\_B)

**Description:** If the condition is true, execution continues at (PC) + displacement. Branches can be forward, with a positive displacement, or backward, with a negative displacement. PC holds the address of the instruction word for the Bcc instruction, plus two.

Condition code specifies one of the following tests, where C, N, V, and Z stand for the condition code bits CCR[C], CCR[N], CCR[V] and CCR[Z], respectively:

Code	Condition	Encoding	Test	Code	Condition	Encoding	Test
CC(HS)	Carry clear	0100	$\bar{C}$	LS	Lower or same	0011	$C \mid Z$
CS(LO)	Carry set	0101	C	LT	Less than	1101	$N \ \& \ \bar{V} \mid \bar{N} \ \& \ V$
EQ	Equal	0111	Z	MI	Minus	1011	N
GE	Greater or equal	1100	$N \ \& \ V \mid \bar{N} \ \& \ \bar{V}$	NE	Not equal	0110	$\bar{Z}$
GT	Greater than	1110	$N \ \& \ V \ \& \ \bar{Z} \mid \bar{N} \ \& \ \bar{V} \ \& \ \bar{Z}$	PL	Plus	1010	$\bar{N}$
HI	High	0010	$\bar{C} \ \& \ \bar{Z}$	VC	Overflow clear	1000	$\bar{V}$
LE	Less or equal	1111	$Z \mid N \ \& \ \bar{V} \mid \bar{N} \ \& \ V$	VS	Overflow set	1001	V

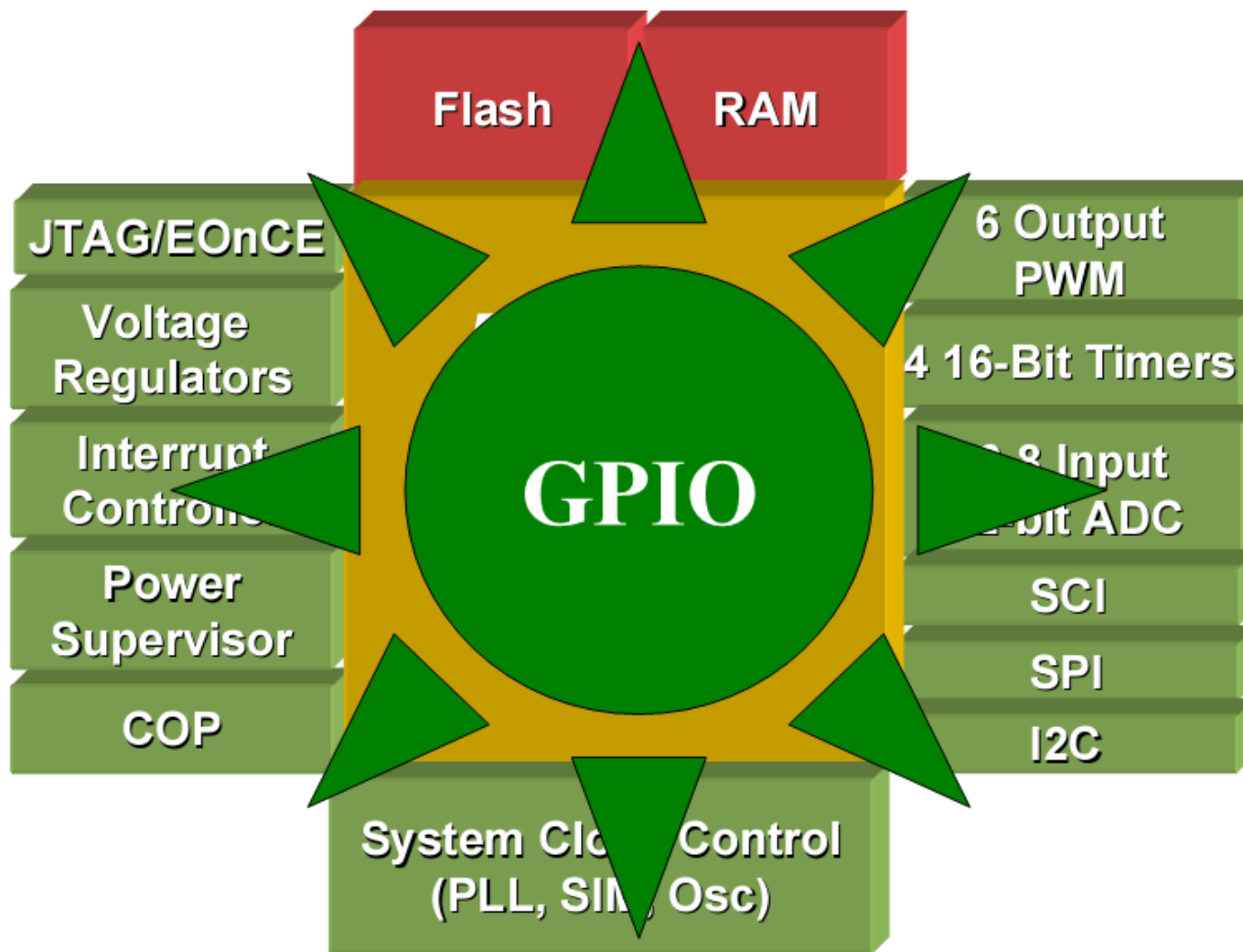
**Condition Codes:** Not affected



# Moduł portów wejścia-wyjścia (General Purpose I/O module)

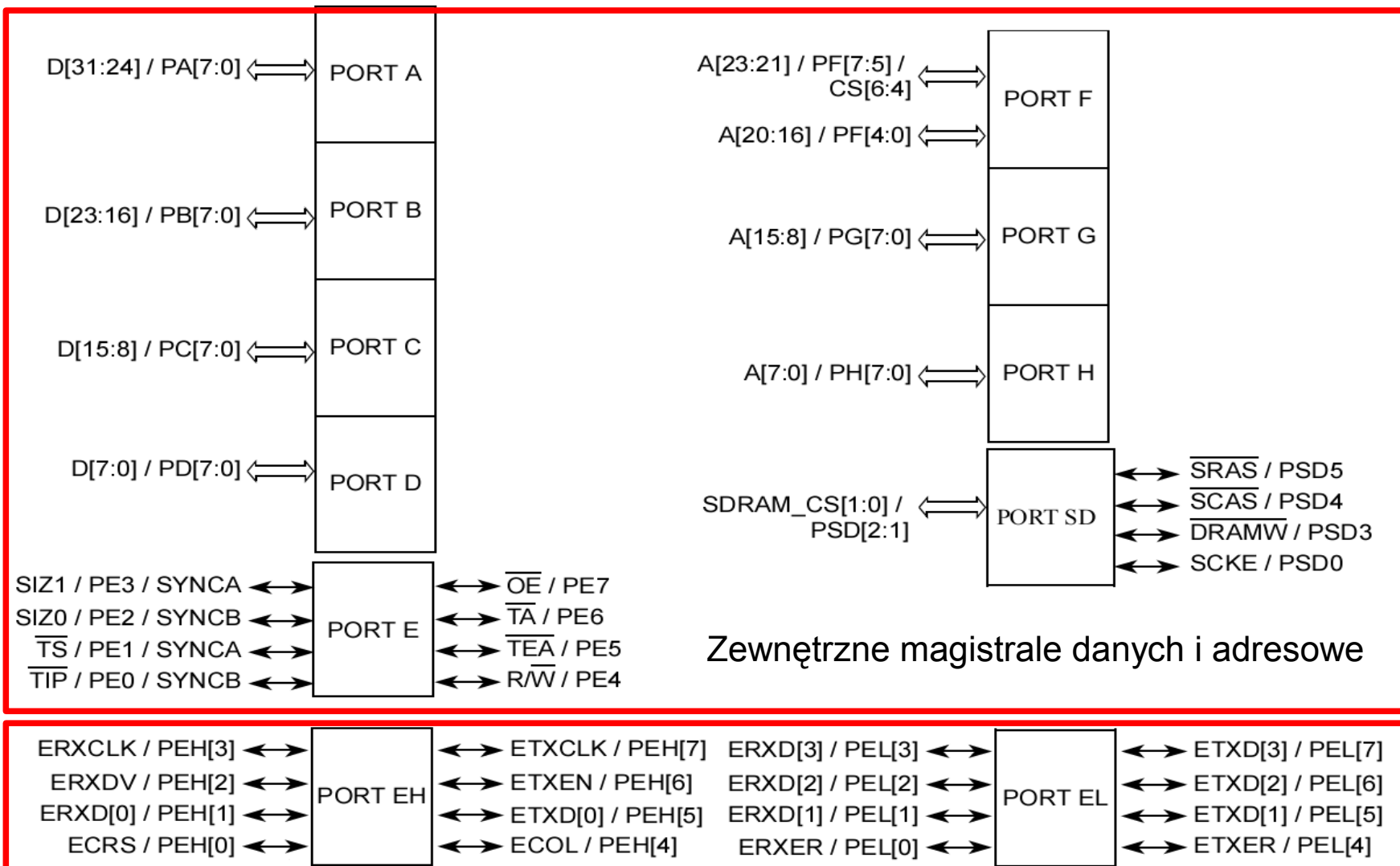


# Moduł portów I/O (1)



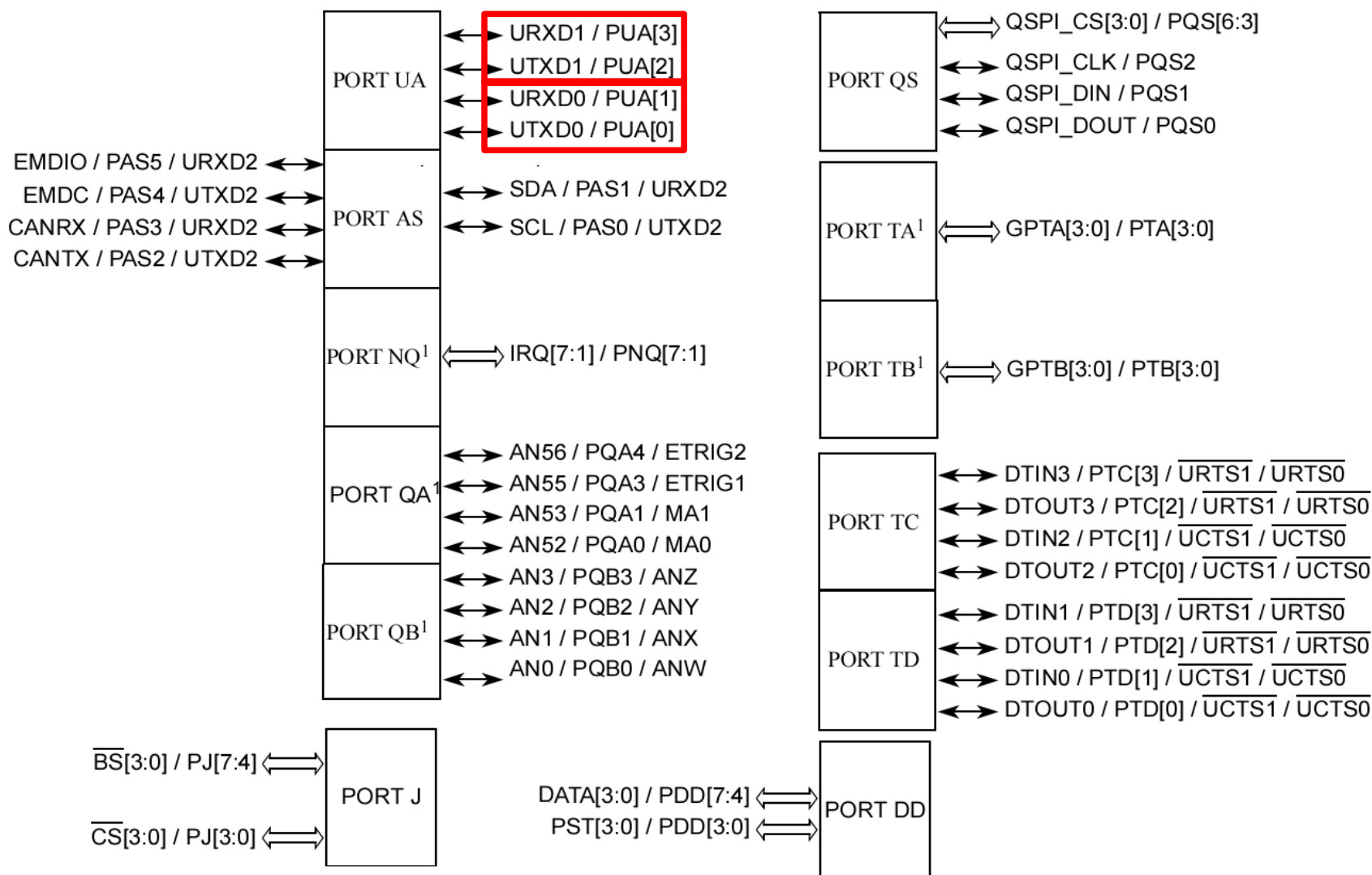


## Moduł portów I/O (2)





# Moduł portów I/O (3)





## Rejestry sterujące

**PnPAR** - rejestr kontrolujący przeznaczenie portu

**DDRn** - Rejestr kontrolujący kierunek sygnałów portu I/O

**PORTn** - rejestr kontrolujący stan wyprowadzeń wyjściowych

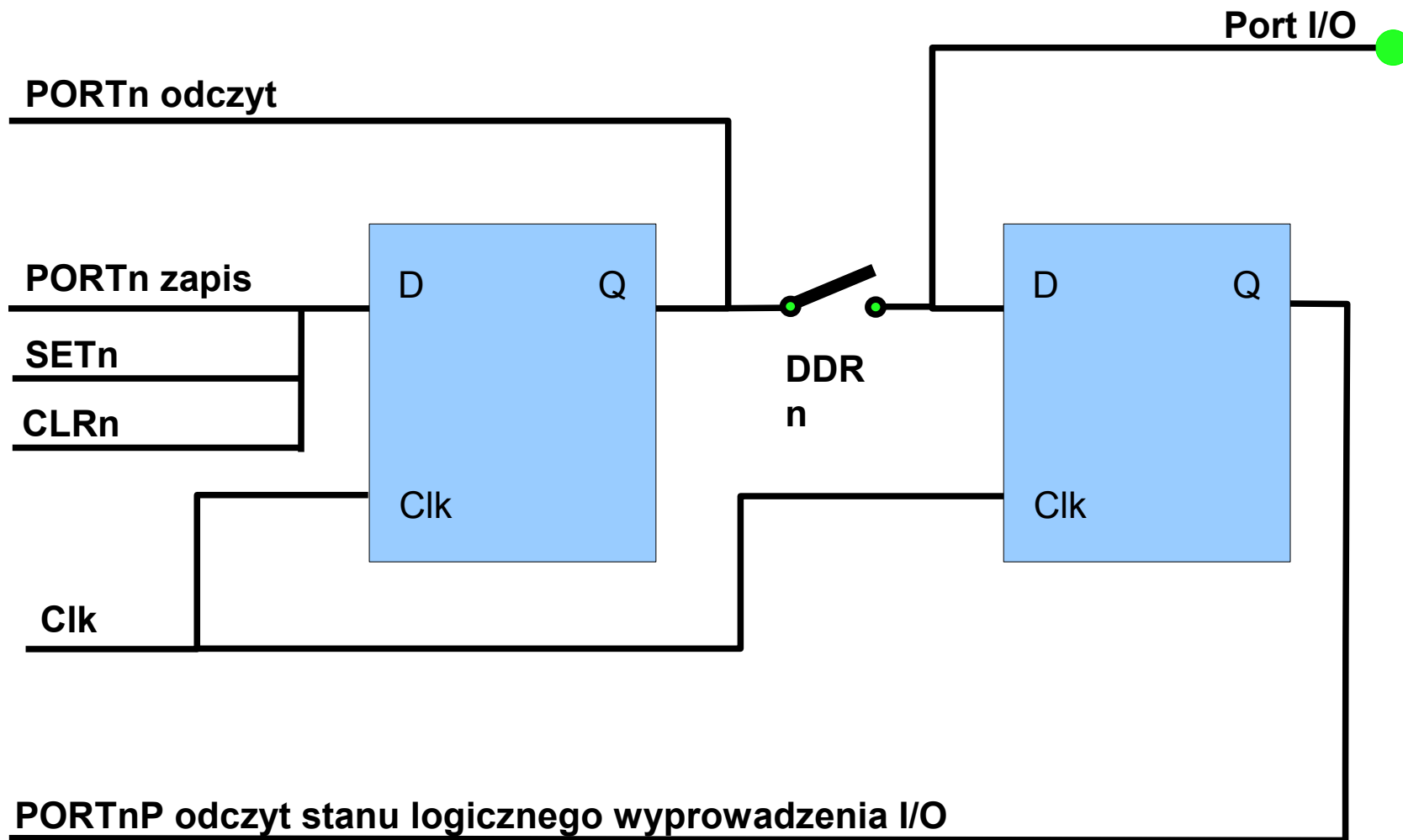
**PORTnP** - rejestr odwzorowujący stan wyprowadzenia I/O

**SETn/CLRn** - Rejestr służący do ustawiania/zerowania przerzutnika wyjściowego





# Schemat blokowy portu I/O





## Przykładowe rejestry sterujące modułu I/O

	7	6	5	4	3	2	1	0
Field	PORTn7	PORTn6	PORTn5	PORTn4	PORTn3	PORTn2	PORTn1	PORTn0
Reset	1111_1111							
R/W:	R/W							
Address	IPSBAR + 0x10_0000 (PORTA), 0x10_0001 (PORTB), 0x10_0002 (PORTC), 0x10_0003 (PORTD), 0x10_0004 (PORTE), 0x10_0005 (PORTF), 0x10_0006 (PORTG), 0x10_0007 (PORTH), 0x10_0008 (PORTJ), 0x10_0009 (PORTDD), 0x10_000A (PORTEH), 0x10_000B (PORTEL)							

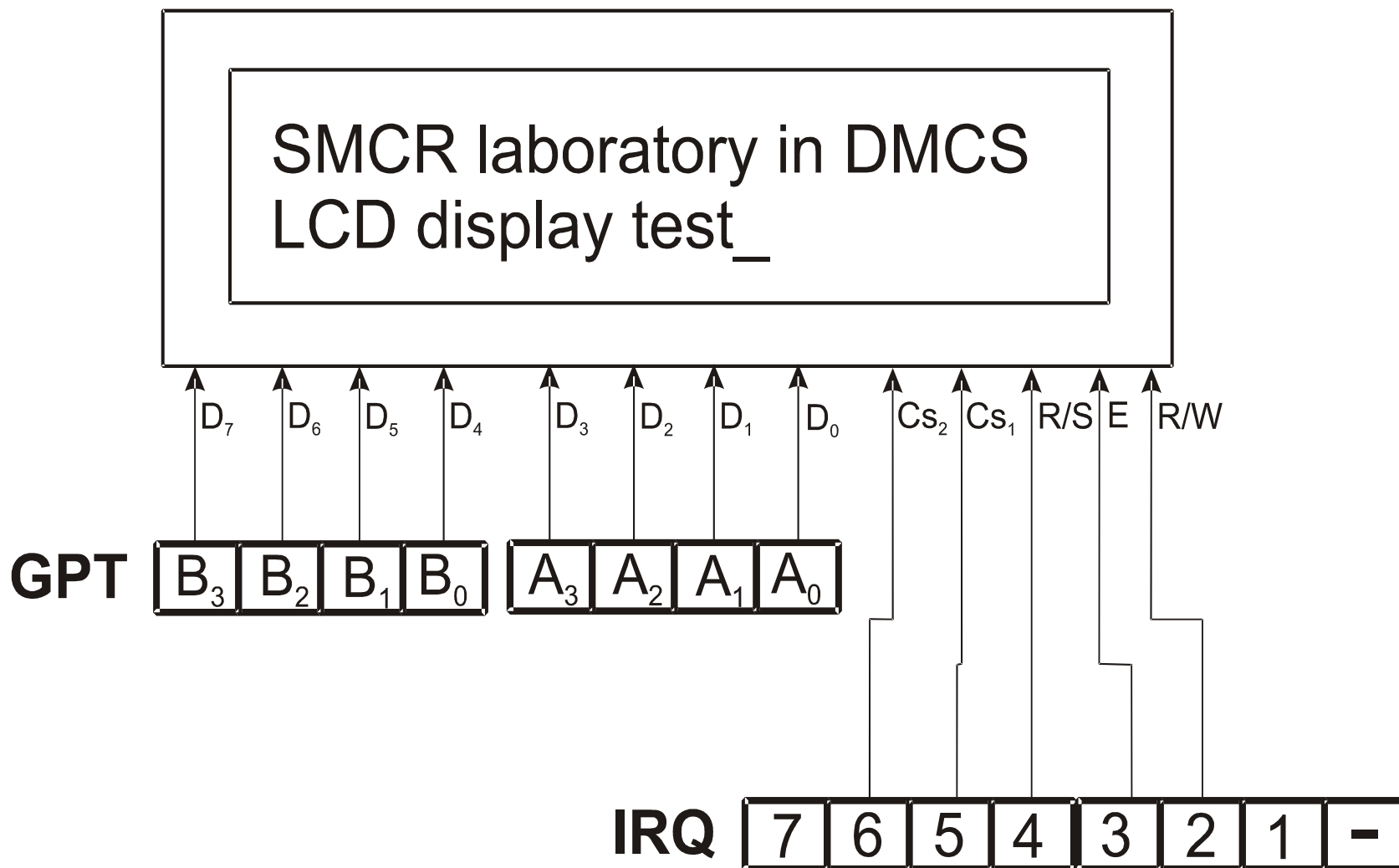
Figure 26-2. Port Output Data Registers (8-bit)

	7	4	3	2	1	0
Field	—		PORTn3	PORTn2	PORTn1	PORTn0
Reset	0000_1111					
R/W:	R			R/W		
Address	IPSBAR + 0x10_000F (PORTTC), 0x10_0010 (PORTTD), 0x10_0011 (PORTUA)					

Figure 26-5. Port Output Data Registers (4-bit)

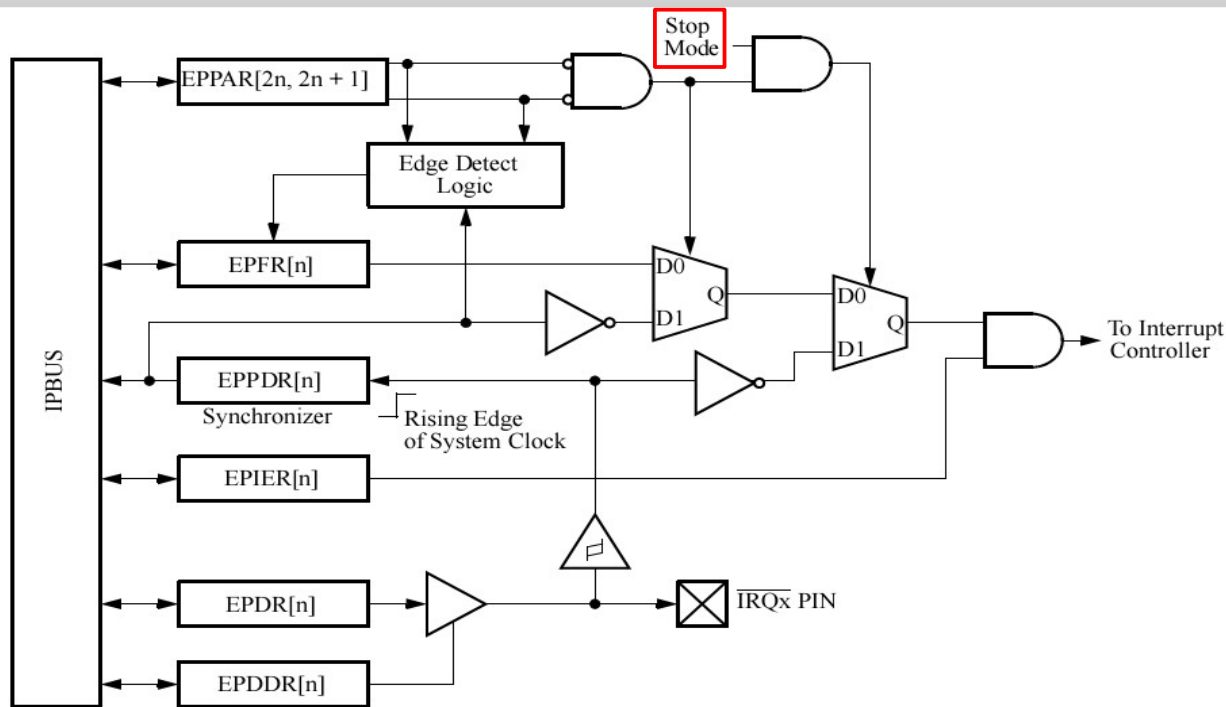


# Podłączenie wyświetlacza LCD





# Port przerwań zewnętrznych (EPORT)



**Table 11-2. Edge Port Module Memory Map**

IPSBAR Offset	Bits 15–8	Bits 7–0	Access <sup>1</sup>
0x0013_0000	EPORT Pin Assignment Register (EPPAR)		S
0x0013_0002	EPORT Data Direction Register (EPDDR)	EPORT Interrupt Enable Register (EPIER)	S
0x0013_0004	EPORT Data Register (EPDR)	EPORT Pin Data Register (EPPDR)	S/U
0x0013_0006	EPORT Flag Register (EPFR)	Reserved <sup>2</sup>	S/U



## Zakresy napięć wejściowych

### Table 33-1. Absolute Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	$V_{DD}$	- 0.3 to +4.0	V
Clock Synthesizer Supply Voltage	$V_{DDPLL}$	- 0.3 to +4.0	V
RAM Memory Standby Supply Voltage	$V_{STBY}$	- 0.3 to + 4.0	V
Flash Memory Supply Voltage	$V_{DDF}$	- 0.3 to +4.0	V
Flash Memory Program / Erase Supply Voltage	$V_{PP}$	- 0.3 to + 6.0	V
Analog Supply Voltage	$V_{DDA}$	- 0.3 to +6.0	V
Analog Reference Supply Voltage	$V_{RH}$	- 0.3 to +6.0	V
Analog ESD Protection Voltage	$V_{DDH}$	- 0.3 to +6.0	V
Digital Input Voltage	$V_{IN}$	- 0.3 to + 6.0	V
Analog Input Voltage	$V_{AIN}$	- 0.3 to + 6.0	V
EXTAL pin voltage	$V_{EXTAL}$	0 to 3.3	V
XTAL pin voltage	$V_{XTAL}$	0 to 3.3	V



## Zakres przedmiotu

- ▶ Systemy mikroprocesorowe, systemy wbudowane
- ▶ **Laboratorium**
- ▶ Rodzina procesorów ARM
- ▶ Urządzenia peryferyjne
- ▶ Pamięci i dekodery adresowe
- ▶ Programy wbudowane na przykładzie procesorów ARM
- ▶ Metodyki projektowania systemów wbudowanych
- ▶ Interfejsy w systemach wbudowanych
- ▶ Systemy czasu rzeczywistego

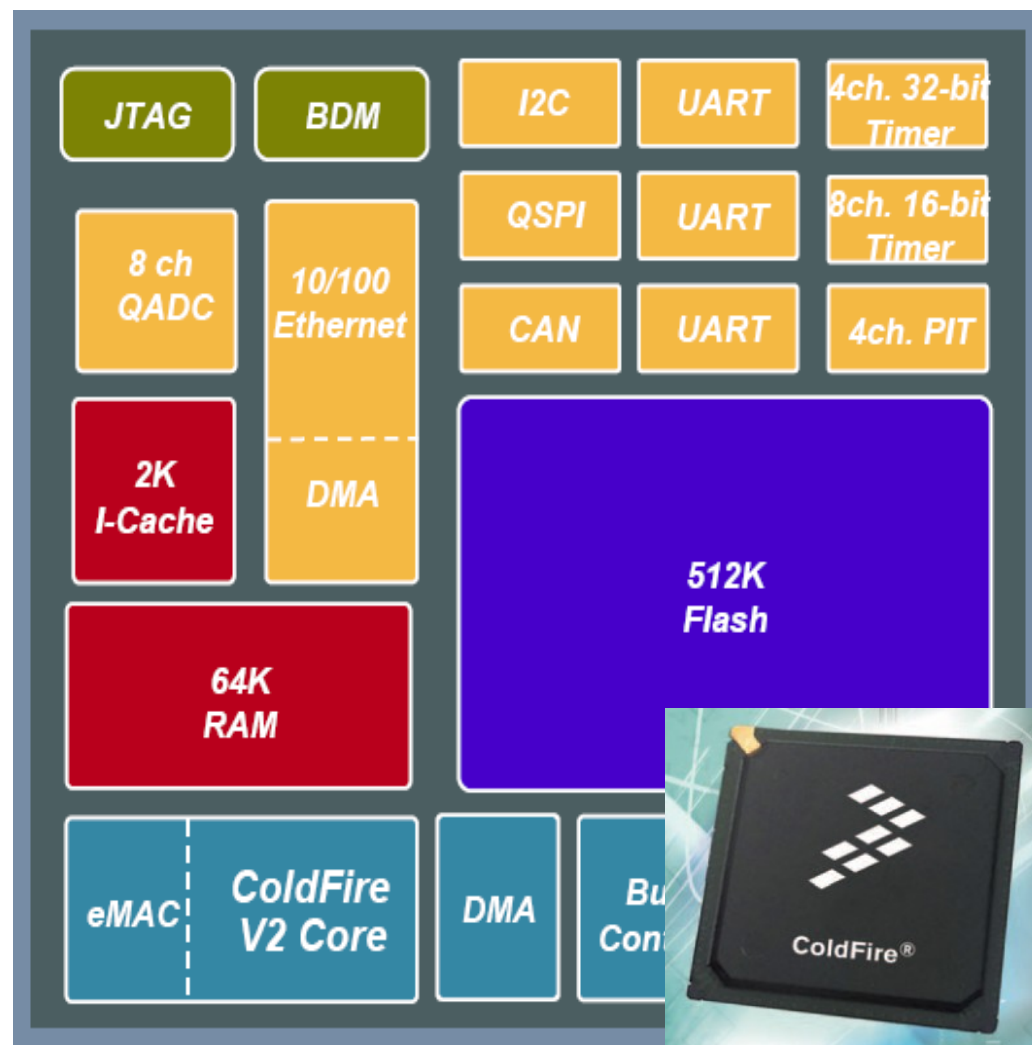


## 32-bitowe procesory firmy Freescale



## Mikrokontroler MCF5282

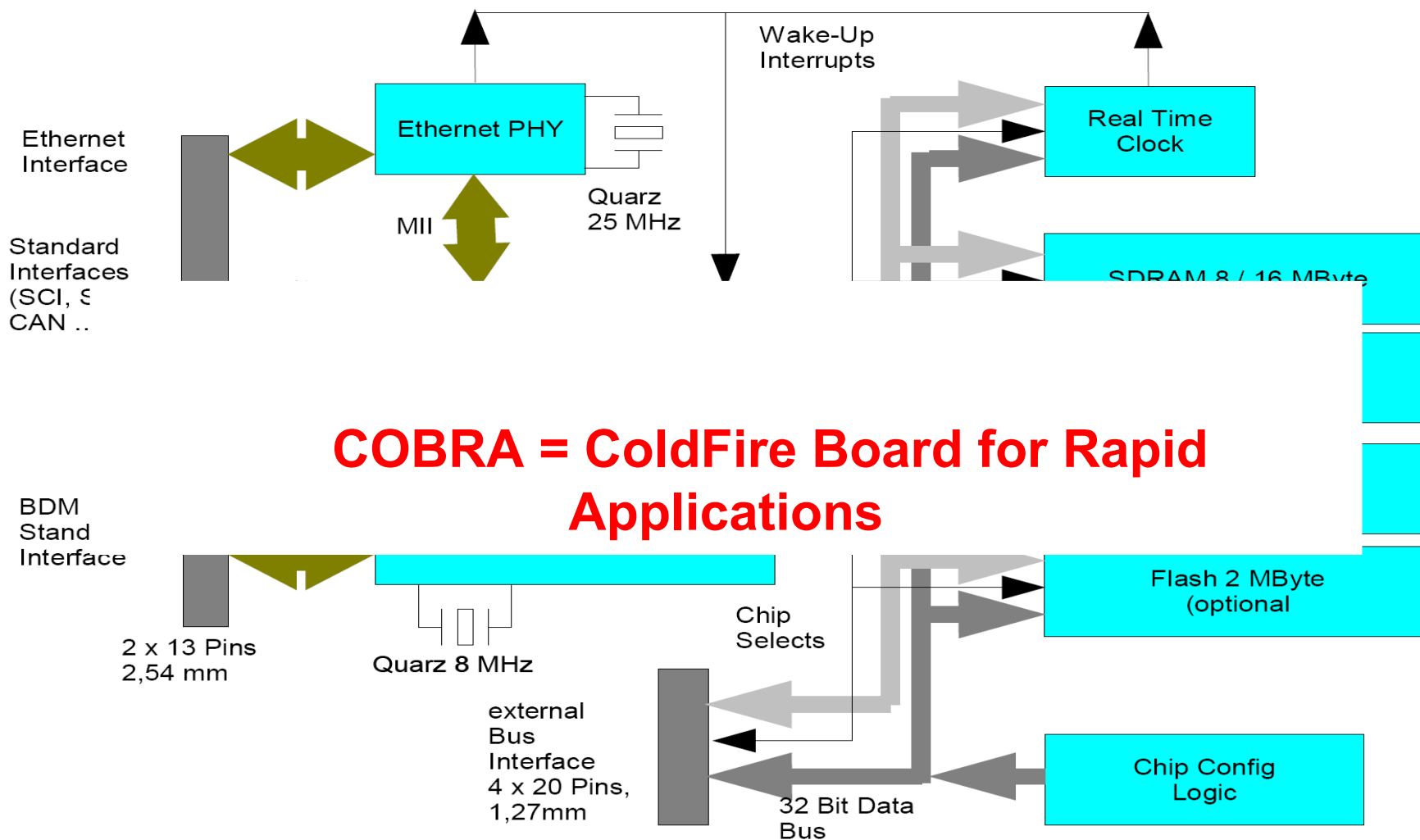
- Maksymalna częstotliwość pracy 80MHz
- 16 32-bitowych rejestrów ogólnego przeznaczenia i adresowych
- 2 k pamięci cache danych lub instrukcji
- 64 k pamięci RAM
- 512 k pamięci Flash
- Tryby pracy z obniżonym poborem mocy (4 tryby pracy)
- Do 142 programowalnych bitowych portów I/O
- Programowalny watch-dog







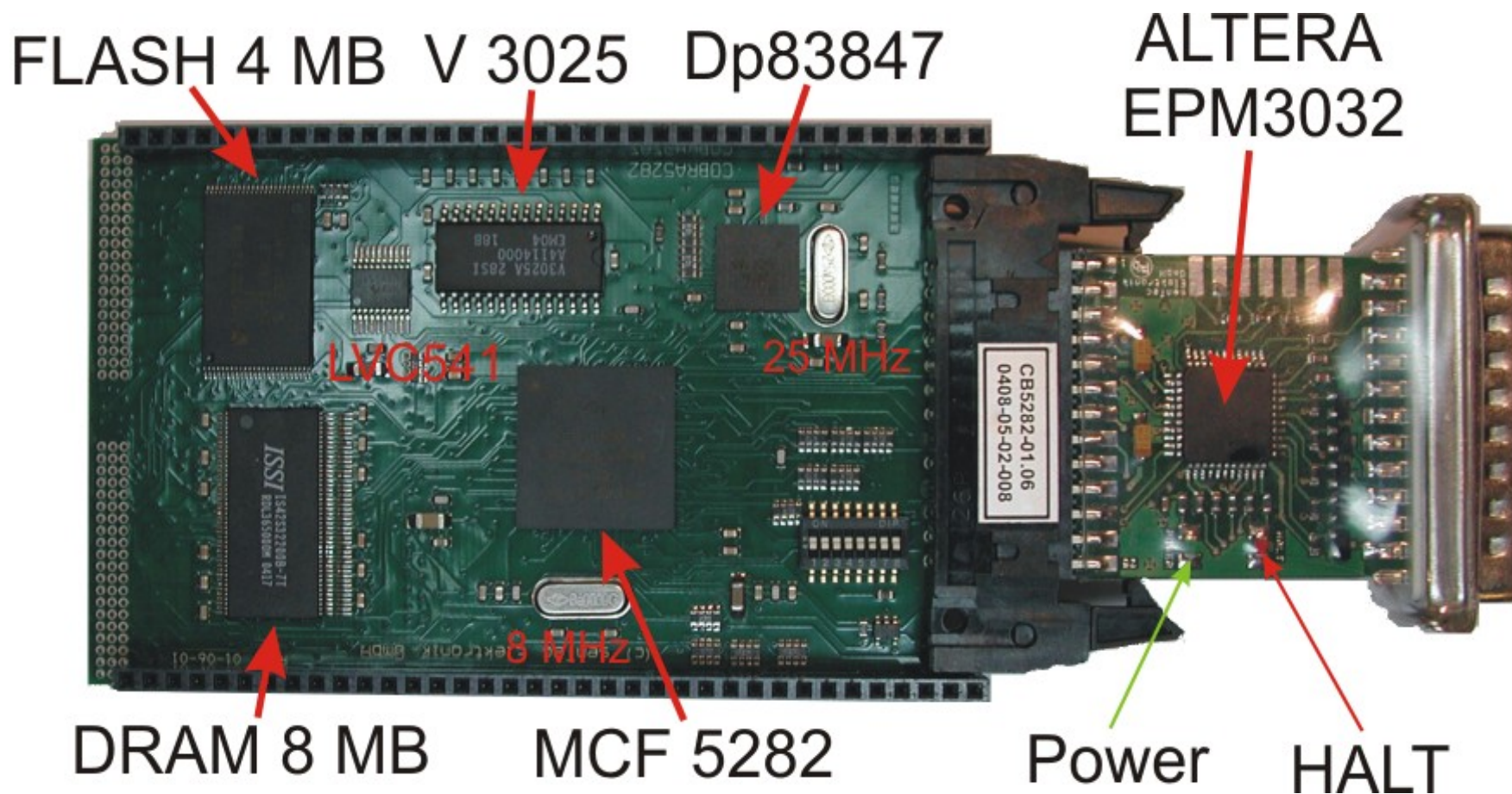
# Schemat blokowy modułu COBRA



**COBRA = ColdFire Board for Rapid Applications**

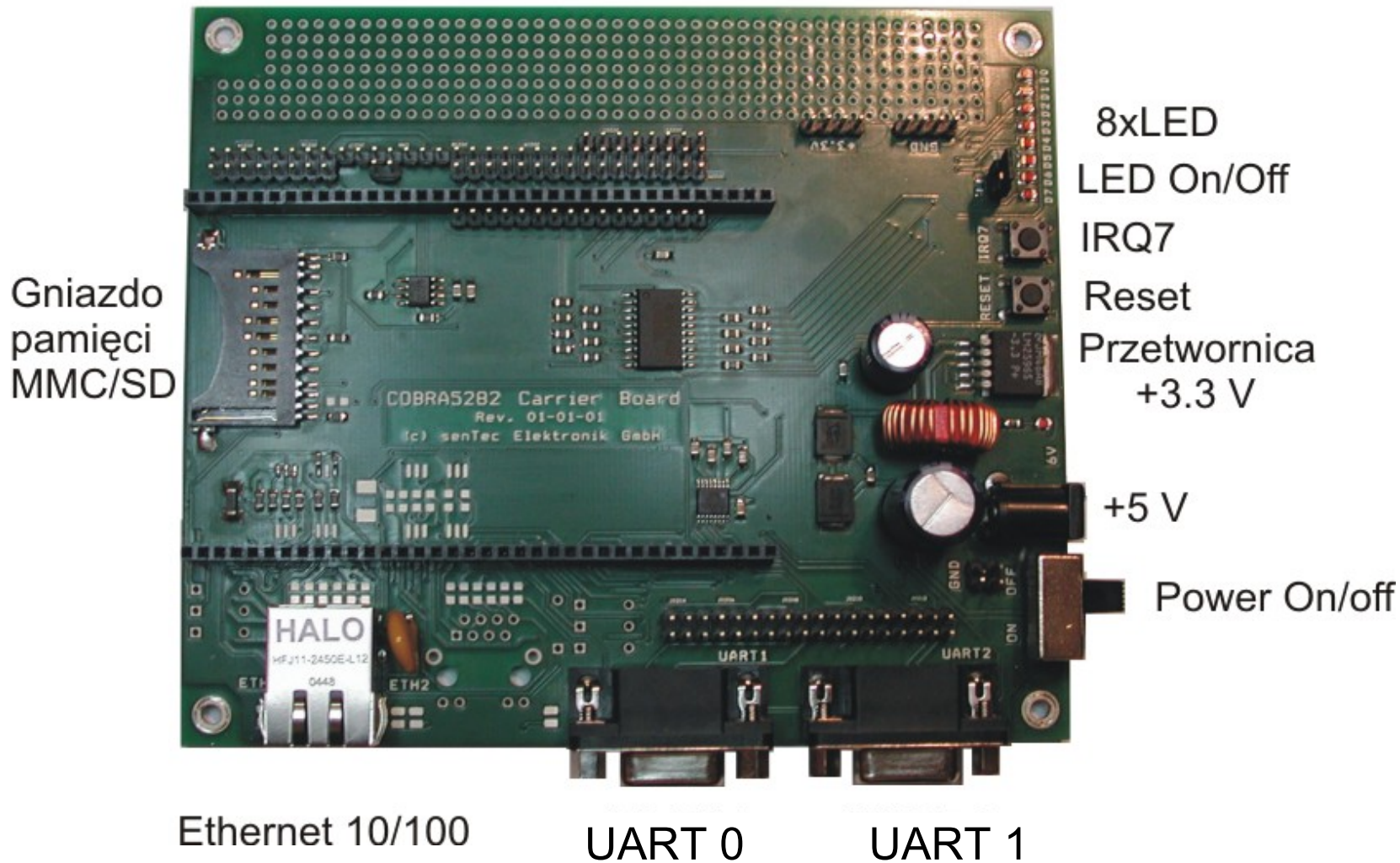


# Moduł COBRA wraz z analizatorem BDM





# Płytki bazowa zestawu uruchomieniowego





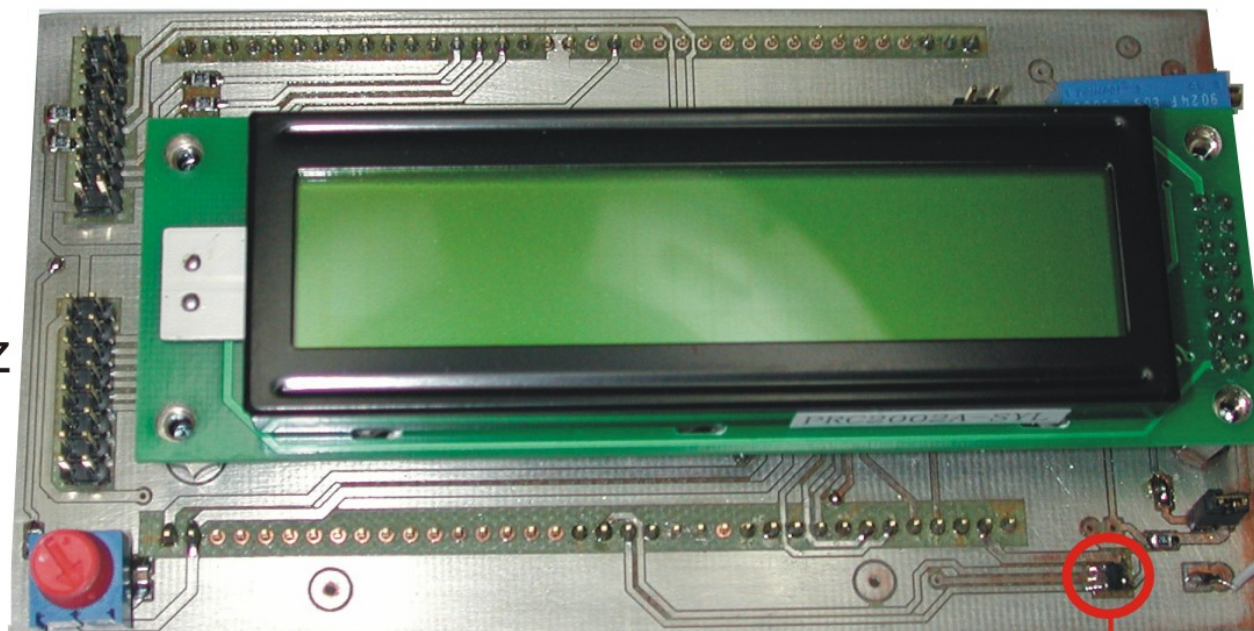
## Płytki z dodatkowymi peryferiami

LCD 2x20 (HDD 44780)

Kontrast LCD

Klawiatura  
3x4

Wyświetlacz  
LED



Potencjometr 0-3,3 V

Termometr SPI  
TMP 123



## Rejestry procesora

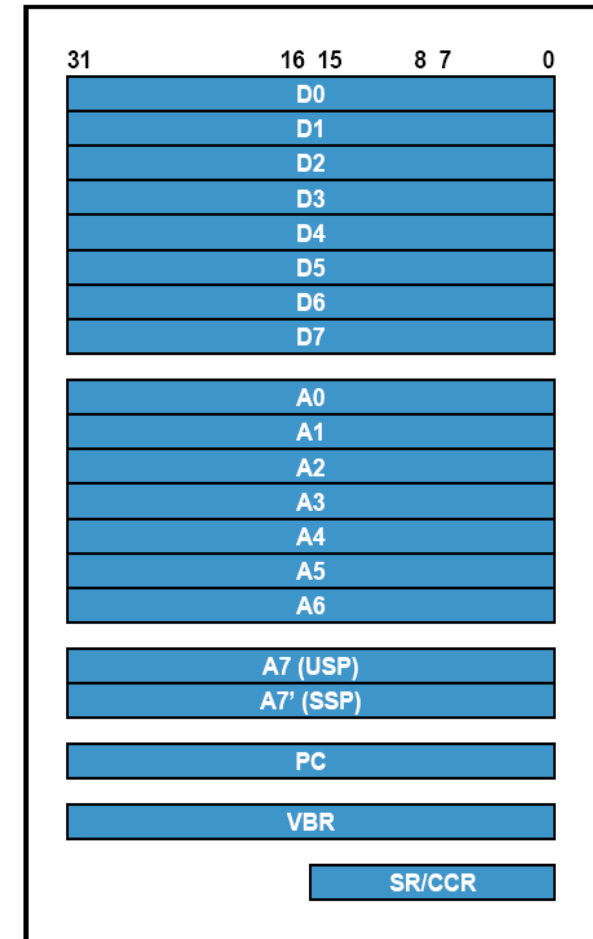
**Rejestry procesora** stanowią komórki wewnętrznej pamięci procesora o niewielkich rozmiarach (najczęściej 4/8/16/32/64/128 bitów) służące do przechowywania tymczasowych wyników obliczeń, adresów danych w pamięci operacyjnej, konfiguracji, itd.

### Cechy rejestrów procesora:

- stanowią najwyższy szczebel w hierarchii pamięci (najszybszy rodzaj pamięci komputera),
- Realizowane w postaci przerzutników dwustanowych,
- Liczba rejestrów zależy od zastosowania procesora.

### Rejestry dzielimy na:

- ★ rejestry danych - do przechowywania danych np. argumentów i wyników obliczeń,
- ★ rejestry adresowe - do przechowywania adresów (wskaźnik stosu, wskaźnik programu, rejestry segmentowe),
- ★ rejestry ogólnego zastosowania (ang. general purpose), przechowują zarówno dane, jak i adresy,
- ★ rejestry zmiennoprzecinkowe - do przechowywania i wykonywania obliczeń na liczbach zmiennoprzecinkowych (koprocessor FPU).



## Rejestry procesora z rodziny Motorola ColdFire



## Charakterystyka jądra ColdFire 2/2M

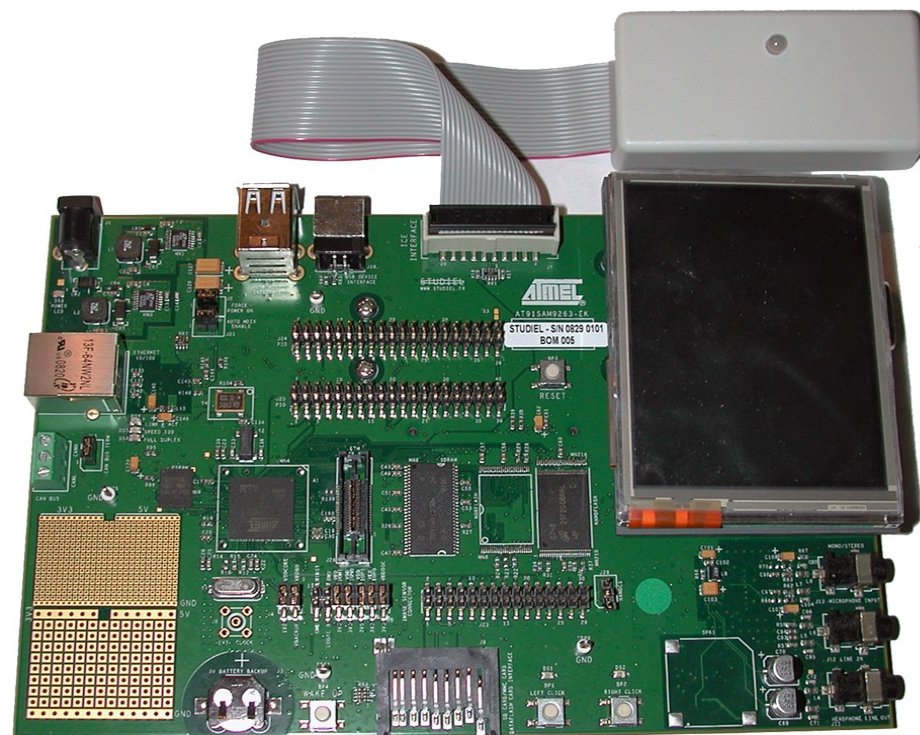
- ▶ 32-bitowa magistrala adresowa (4 GB)
- ▶ 32-bitowa magistrala danych
- ▶ Rdzeń procesora RISC o zmiennej długości rozkazów (16-, 32- oraz 48 bitowe słowa)
- ▶ Procesor posiada zoptymalizowaną pod względem wydajności listę rozkazów bazującą na procesorze rodziny Freescale (Motorola) 68k
- ▶ Lista rozkazów zoptymalizowana pod kątem języków wysokiego poziomu
- ▶ 16 rejestrów ogólnego przeznaczenia (D0 – D7, A0 – A7)
- ▶ Jednostka eMAC (Multiply Accumulate) przyspieszająca obliczenia stałoprzecinkowe (tylko ColdFire2M)
- ▶ Tryb pracy dedykowanego użytkownika Supervisor mode umożliwiający podniesienie bezpieczeństwa pracy systemów operacyjnych
- ▶ Sterownik zewnętrznej pamięci SRAM/DRAM
- ▶ Mechanizm podręcznej pamięci Cache
- ▶ 12 trybów adresowania pamięci zgodnych z rodziną Freescale (Motorola) 68k
- ▶ Pełne wsparcie debugowania programu w czasie rzeczywistym



# Procesory ARM firmy ATMEL

## Zestaw ewaluacyjny firmy MSC (1)

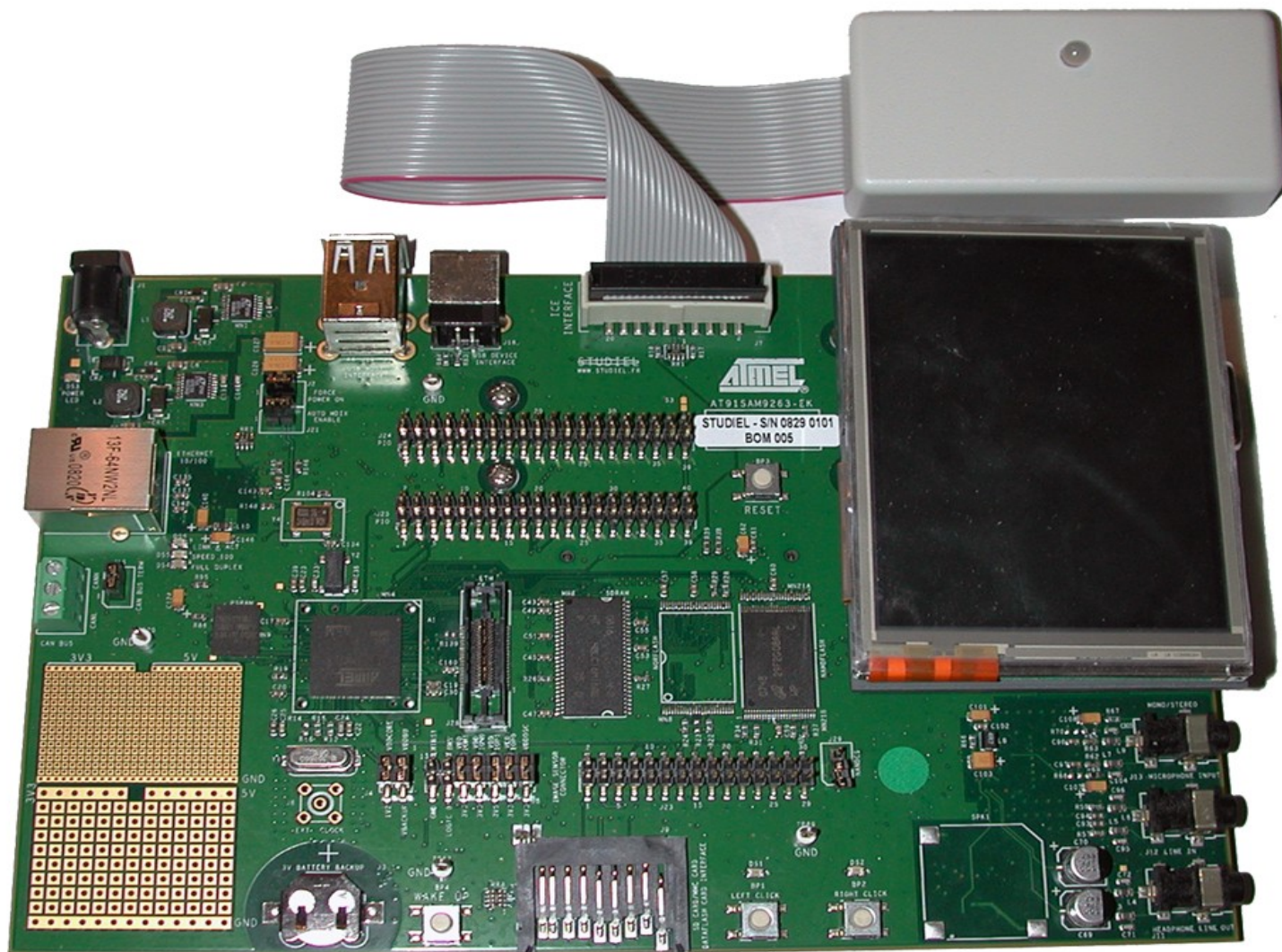
- **Procesor z rdzeniem ARM9TDMI firmy ATMEL: AT91SAM9263**
- **Dostępne pamięci: 64 MB SDRAM, 256 MB NAND FLASH, 4 MB DataFlash, FlashCard slots**
- **Dostępne interfejsy: Ethernet 100-base TX, USB FS device, 2 x USB FS Host, CAN 2.0B, EIA RS232,**
- **Wyświetlacz: 3.5" 1/4 VGA TFT LCD z ekranem dotykowym**
- **Kodek audio: AC97 Audio DAC**
- **Debug interfece: JTAG**
- **Interfejs do programowania: JTAG, Free Atmel SAM-BA tools**
- **Złącze: SD/SDIO/MMC card slot**
- **Oznaczenie producenta: AT91SAM9263-STARTUP-PAKET**





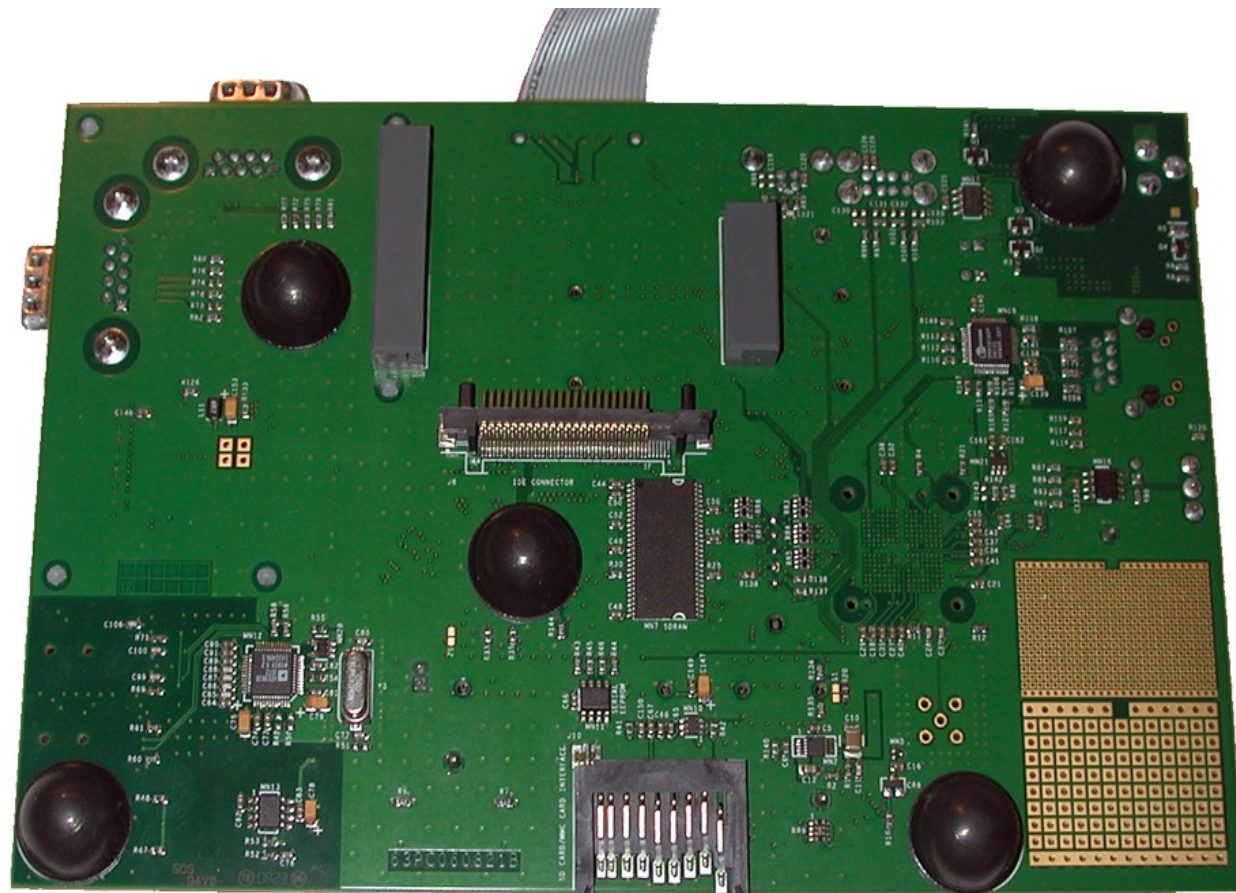
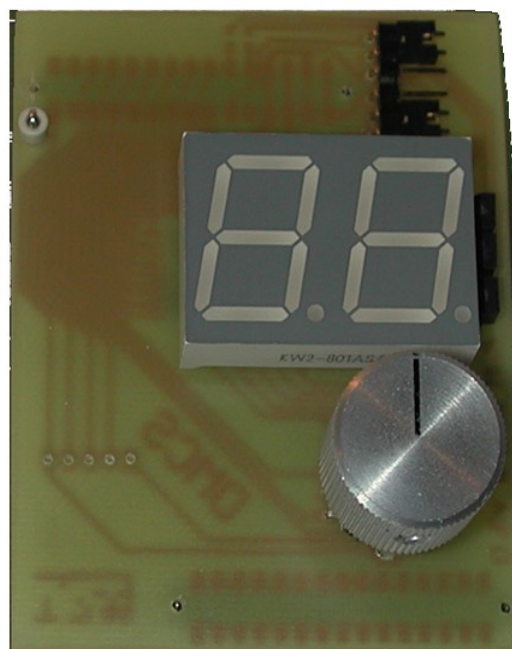


## Zestaw ewaluacyjny firmy MSC (2)





# Zestaw ewaluacyjny firmy MSC (3)





## Narzędzia GNU

**GNU ARM toolchain** – narzędzia dla procesorów z rodziny ARM dostępne w ramach licencji GNU GPL (General Public License).

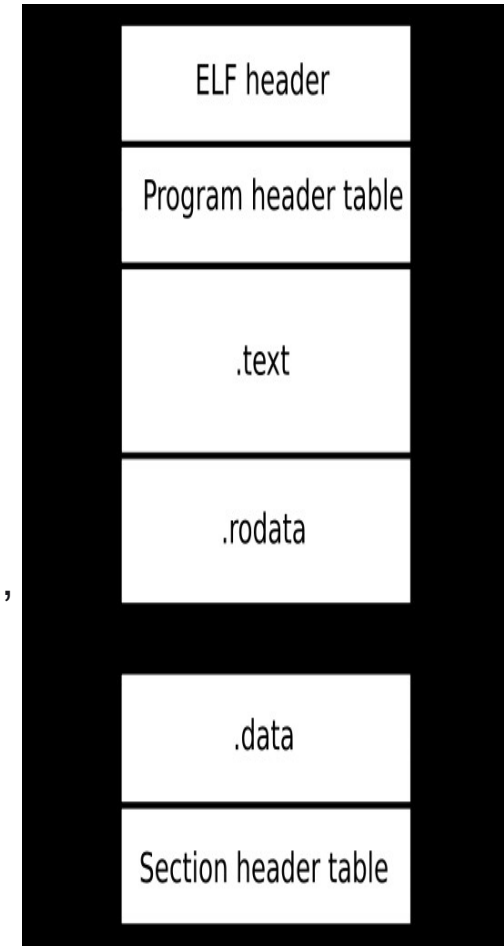
Obecnie dostępne narzędzia (<http://www.gnuarm.org/>):

- ◆ GCC-4.3 toolchain (Linux):
  - ◆ Kompilator: gcc-4.3.2
  - ◆ Przydatne narzędzia: binutils-2.19
  - ◆ Biblioteki C/C++: newlib-1.16
  - ◆ Debugger zgodny z GDB: insight-6.8
- ◆ Cygwin (Windows):
  - ◆ binutils-2.19, gcc-4.3.2-c-c++, newlib-1.16.0, insight-6.8, setup.exe
- ◆ Debugger JTAG z interfejsem USB (informacje dostępne w Katedrze DMCS)



## COFF vs ELF

- **COFF (Common Object File Format)** – standard plików wykonywalnych, relokowalnych i bibliotek dynamicznych opracowany na potrzeby systemów operacyjnych bazujących na systemie Unix. COFF miał zastąpić standard plików **a.out**. Wykorzystywany na różnych systemach, również Windows. Obecnie standard COFF wypierany jest przez pliki ELF.
- **ELF (Executable and Linkable Format)** – standard plików wykonywalnych, relokowalnych, bibliotek dynamicznych i zrzutów pamięci wykorzystywany na różnych komputerach i systemach operacyjnych, np.: rodzina x86, PowerPC, OpenVMS, BeOS, konsole PlayStation Portable, PlayStation 2, PlayStation 3, Wii, Nintendo DS, GP2X, AmigaOS 4 oraz Symbian OS v9.
- Przydatne narzędzia:
  - readelf
  - elfdump
  - objdump



Źródło: wikipedia



## Wnętrza plików...

**file** at91sam9263\_getting\_started\_sdram.elf

at91sam9263\_getting\_started\_sdram.elf: ELF 32-bit LSB executable, ARM, version 1, statically linked, not stripped

**file** main.o

main.o: ELF 32-bit LSB relocatable, ARM, version 1, not stripped

**arm-elf-objdump** -d at91sam9263\_getting\_started\_sdram.elf | less

at91sam9263\_getting\_started\_sdram.elf: file format elf32-littlearm

Disassembly of section .text:

20000000 <\_stext>:

20000000: e59ff09c ldr pc, [pc, #156] ; 200000a4 <\_lp\_data+0xc>

20000004 <\_low\_level\_init>:

20000004: e59f209c ldr r2, [pc, #156] ; 200000a8 <\_lp\_data+0x10>

20000008: e8920003 ldm r2, {r0, r1}

2000000c: e1a0d001 mov sp, r1

20000010: e1a0e00f mov lr, pc

20000014: e12fff10 bx r0



# Rejestry procesora a GDB

## (gdb) info r

r0	0x2	0x2
r1	0x20000ba4	0x20000ba4
r2	0x57b	0x57b
r3	0x270f	0x270f
r4	0x300069	0x300069
r5	0x3122dc	0x3122dc
r6	0x1000	0x1000
r7	0x800bc004	0x800bc004
r8	0x3122c4	0x3122c4
r9	0x407c81a4	0x407c81a4
r10	0x441029ab	0x441029ab
r11	0x313f2c	0x313f2c
r12	0x313f30	0x313f30
sp	0x313f18	0x313f18
lr	0x20000a7c	0x20000a7c
pc	0x20000474	0x20000474 <delay+60>
fps	0x0	0x0
cpsr	0x80000053	0x80000053

r0
r1
r2
r3
r4
r5
r6
r7
r8
r9
r10
r11
r12
r13 (sp)
r14 (lr)
r15 (pc)
cpsr