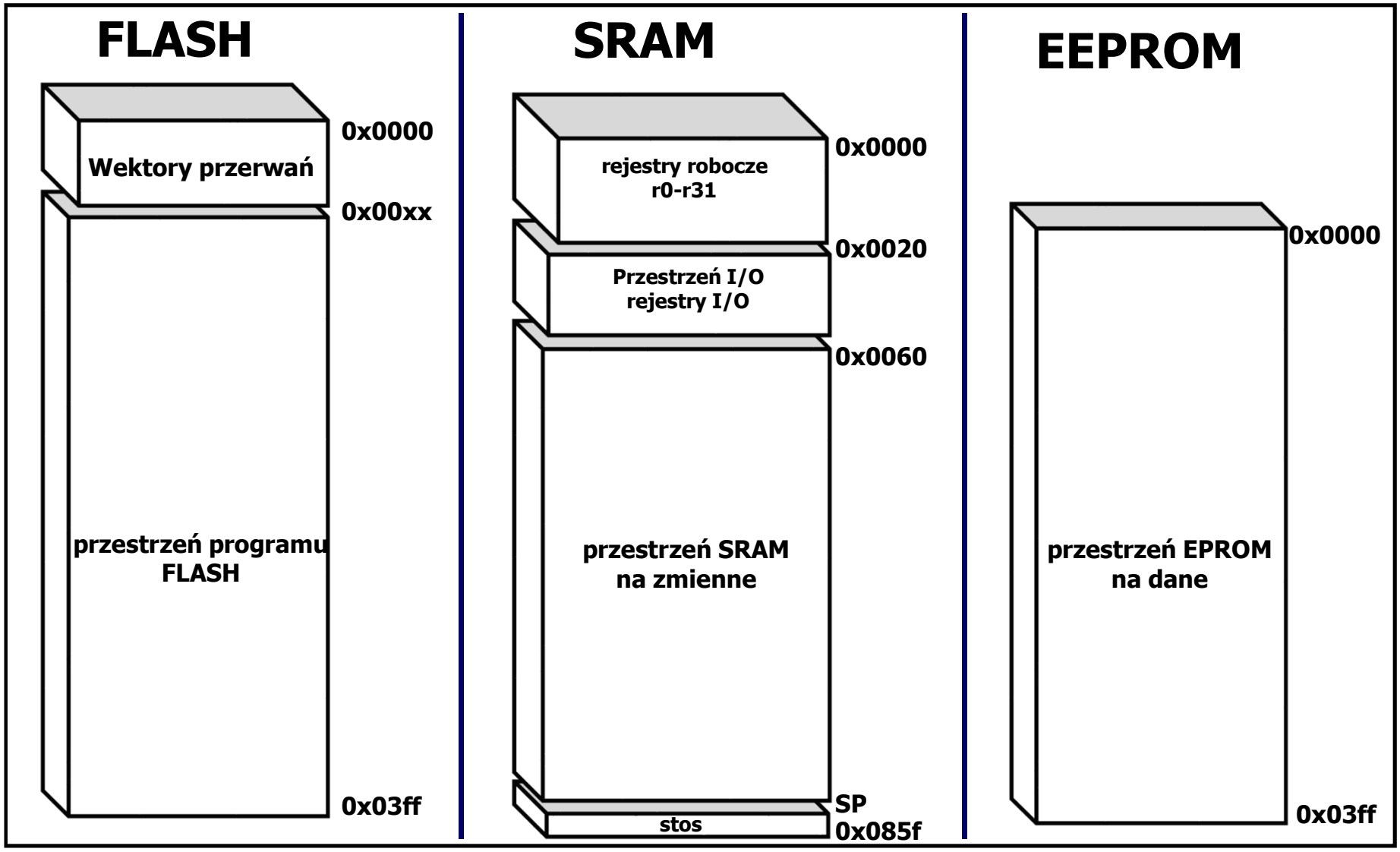

Asembler
4

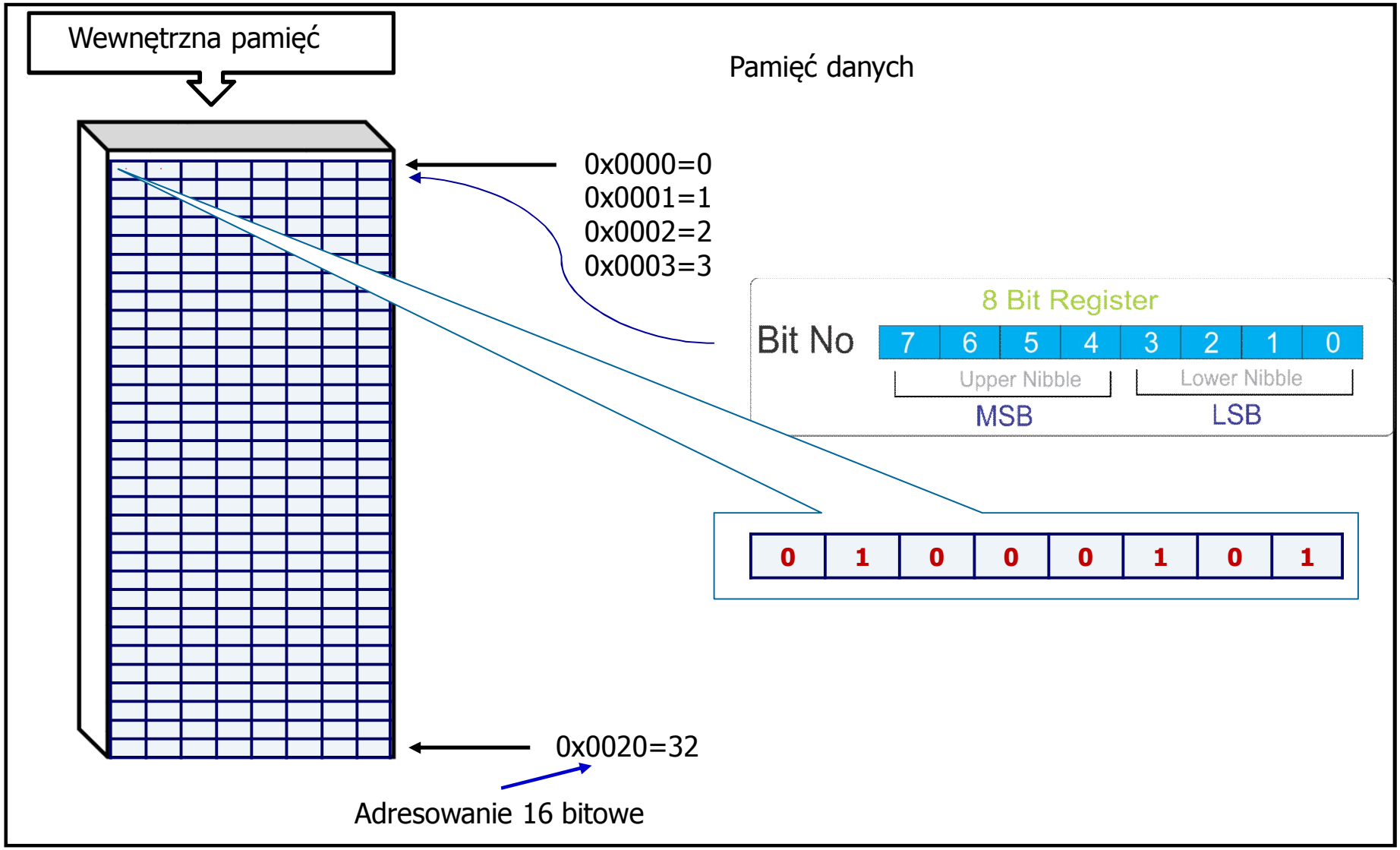
WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Podział pamięci mikrokontrolera



WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Przestrzeń pamięci danych



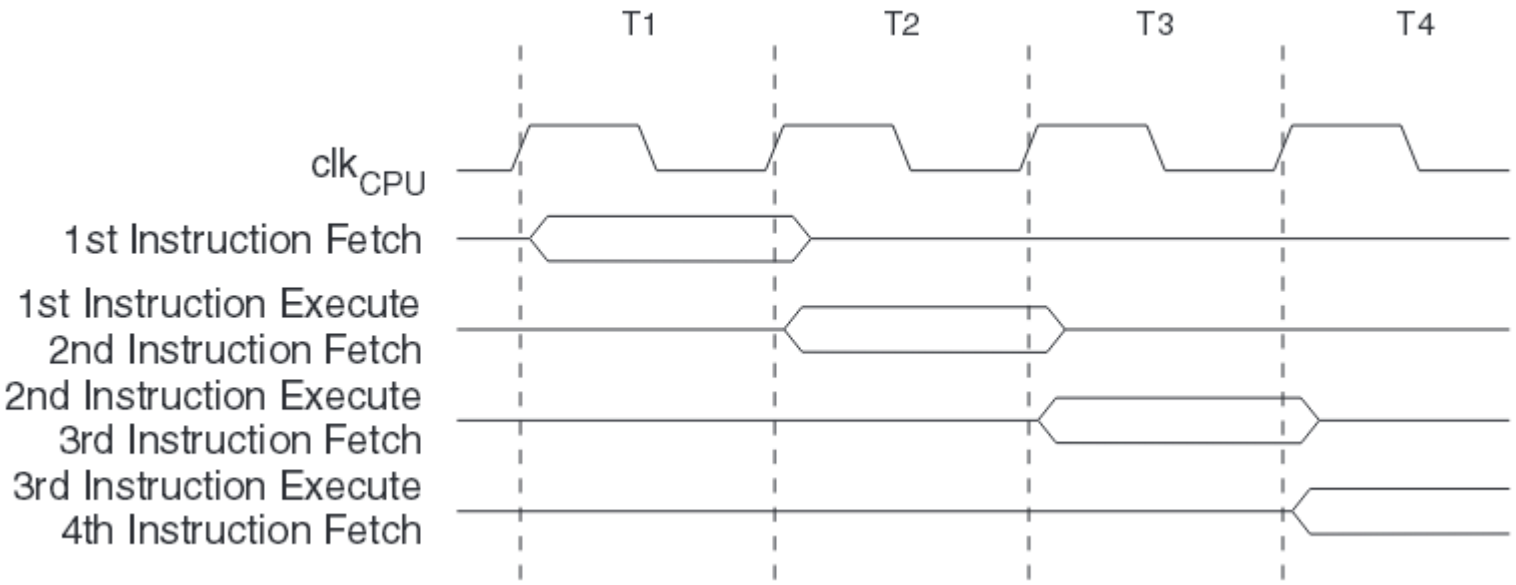
Przestrzeń adresowa pamięci programu

Obszar pamięci programu w układach AVR może być traktowany jako jednolita przestrzeń adresowa pod warunkiem, że nie używamy *boot loadera*.

Pamięć Programu *z boot loaderem*

FLASHEND = 0x3fff
FLASHEND = 16383=16KB

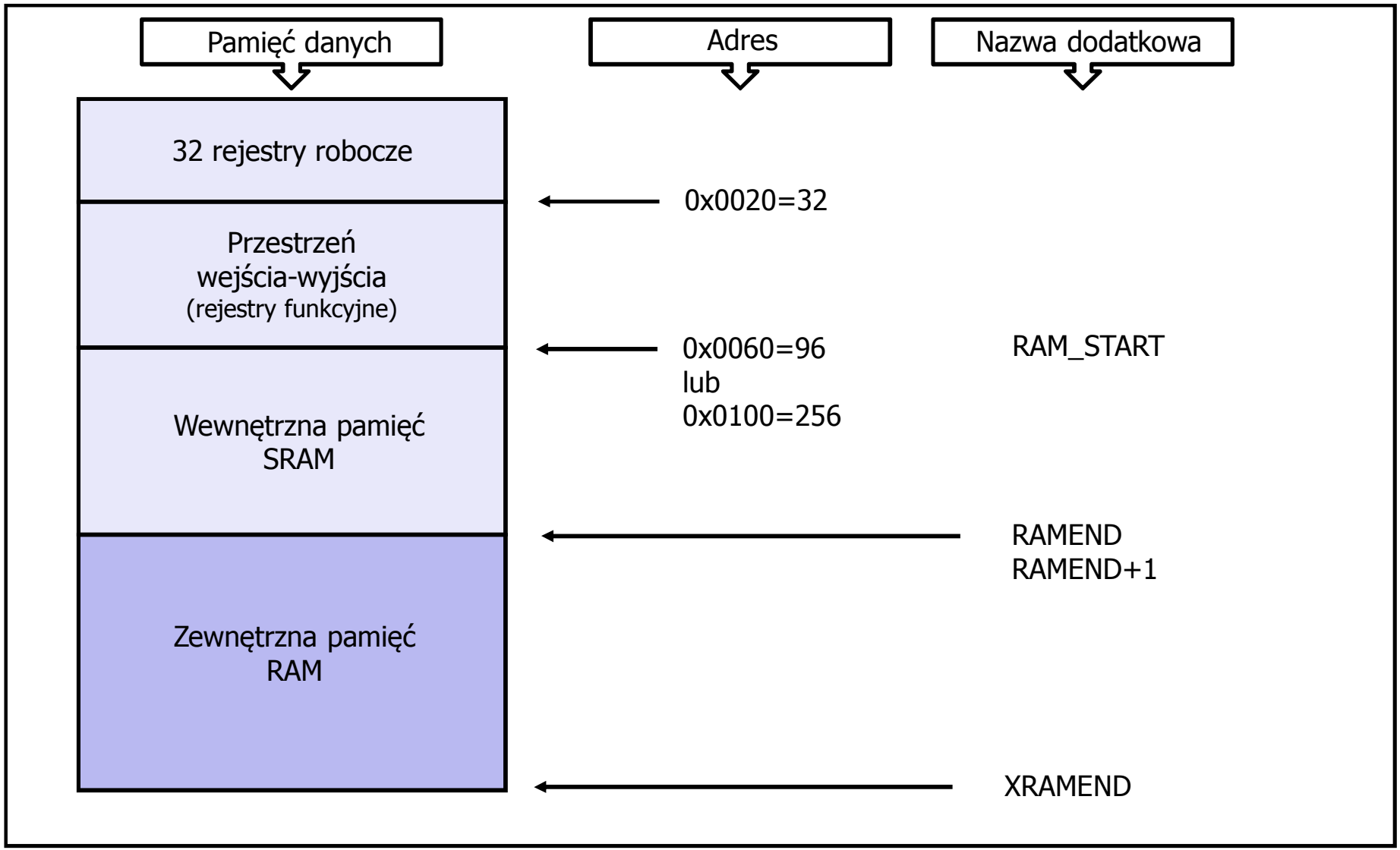
Przetwarzanie potokowe (pipelining)



W AVR zastosowano przetwarzanie potokowe które polega na jednoczesnym wykonywaniu danej instrukcji i pobieraniu instrukcji następnej.
Ma to miejsce w jednym cyklu zegarowym.
AVR są układami 8 bitowymi jednak ich słowo rozkazowe ma 16 bitów.

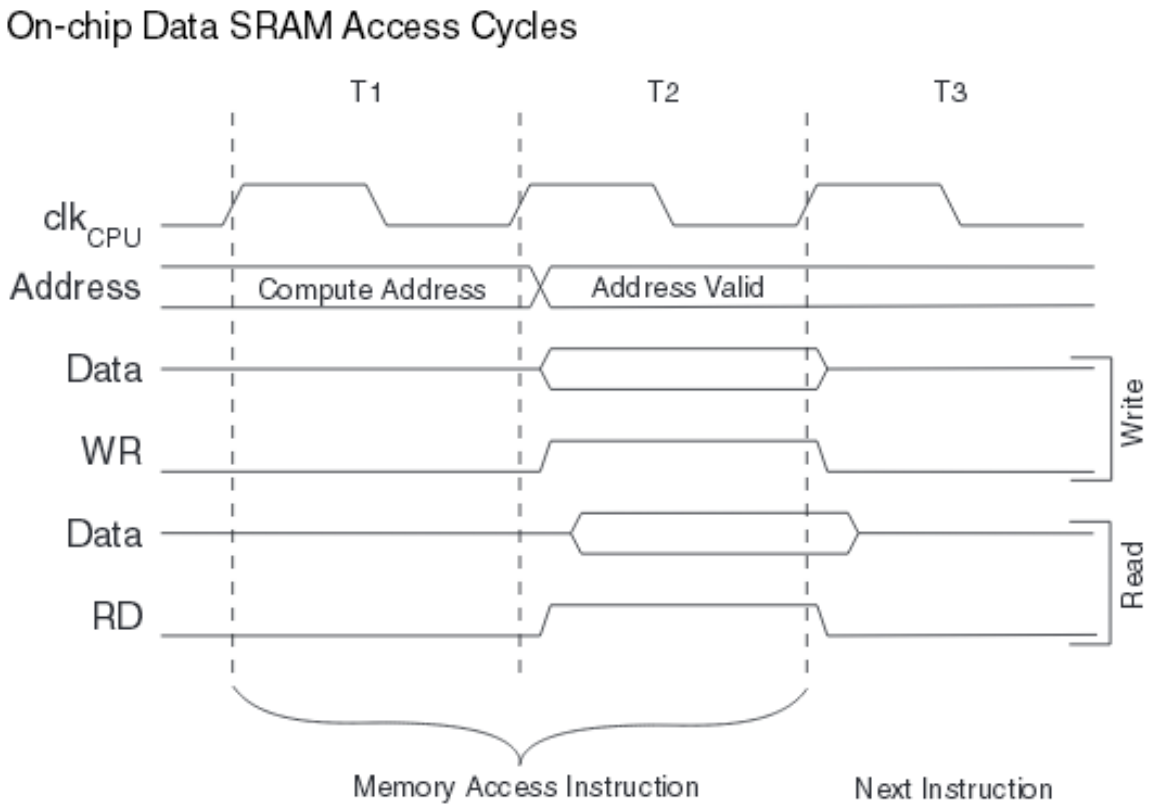
Dla zegara 1MHz można osiągnąć 1 MIPS

Przestrzeń adresowa pamięci danych

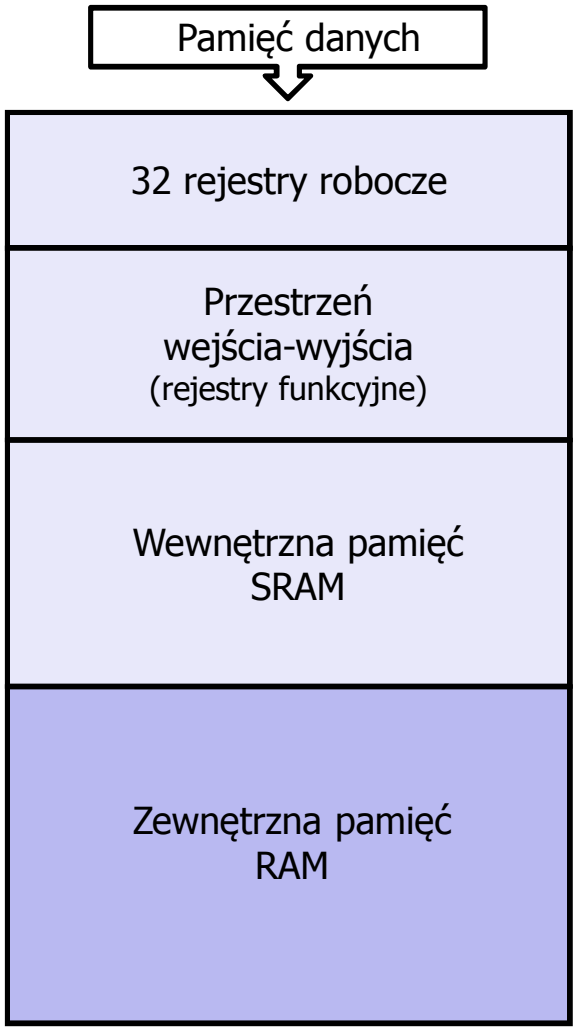


Przestrzeń adresowa pamięci danych

Dostęp do pamięci SRAM



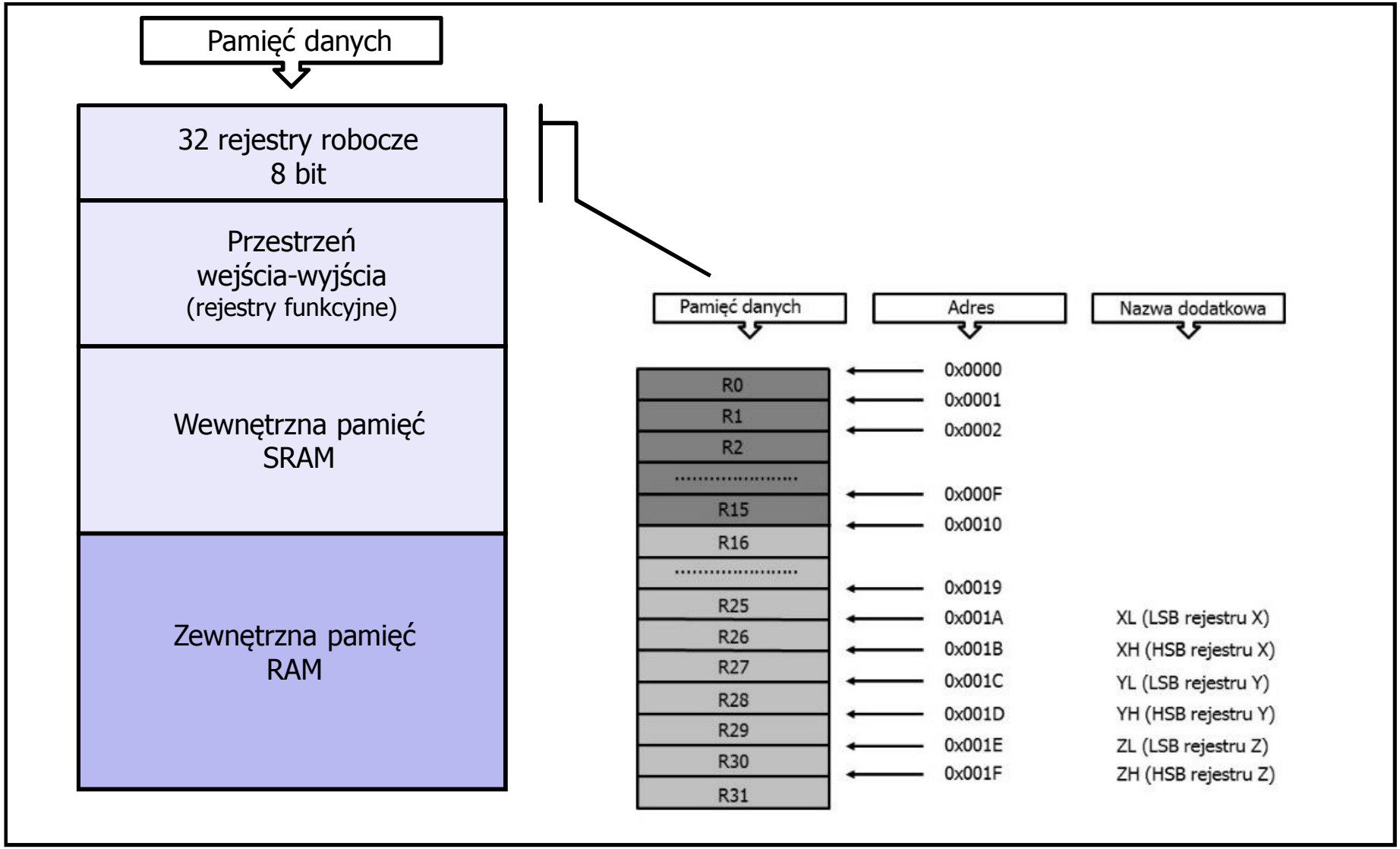
Przestrzeń adresowa pamięci danych



Włączony do kompilacji plik .include "m32def.inc"
Informacje o pamięci zapisane w listingu kompilacji

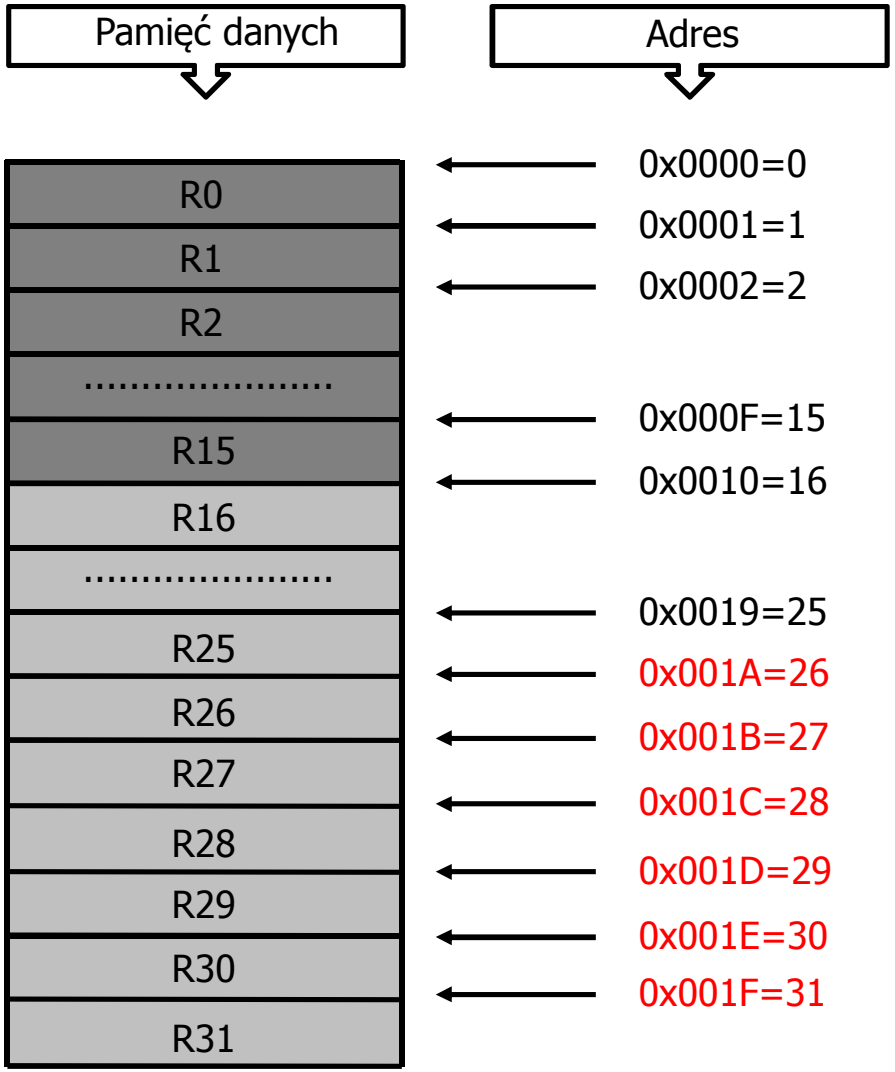
```
; ***** DATA MEMORY DECLARATIONS **************************

.equ    FLASHEND    = 0x3fff    16383
.equ    IOEND       = 0x003f    63
.equ    SRAM_START  = 0x0060    256
.equ    SRAM_SIZE   = 2048      2048
.equ    RAMEND      = 0x085f    2143
.equ    XRAMEND     = 0x0000
.equ    E2END       = 0x03ff
.equ    EEPROMEND   = 0x03ff
.equ    EEADRBITS   = 10

#pragma AVRPART MEMORY PROG_FLASH 32768
#pragma AVRPART MEMORY EEPROM 1024
#pragma AVRPART MEMORY INT_SRAM SIZE 2048
#pragma AVRPART MEMORY INT_SRAM START_ADDR 0x60
```

32 rejestry robocze



WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

32 rejestry robocze



UWAGA !!!!!!

Rejestry R0 … R15 zostały objęte ograniczeniem polegającym na tym, że nie mogą być one używane przez ALU przy operacjach ze stałymi ładowanymi bezpośrednio (immediate). Rozkazy, których nie możemy używać z tymi rejestrami to:

ldi
subi
sbci
sndi
ori
cpi

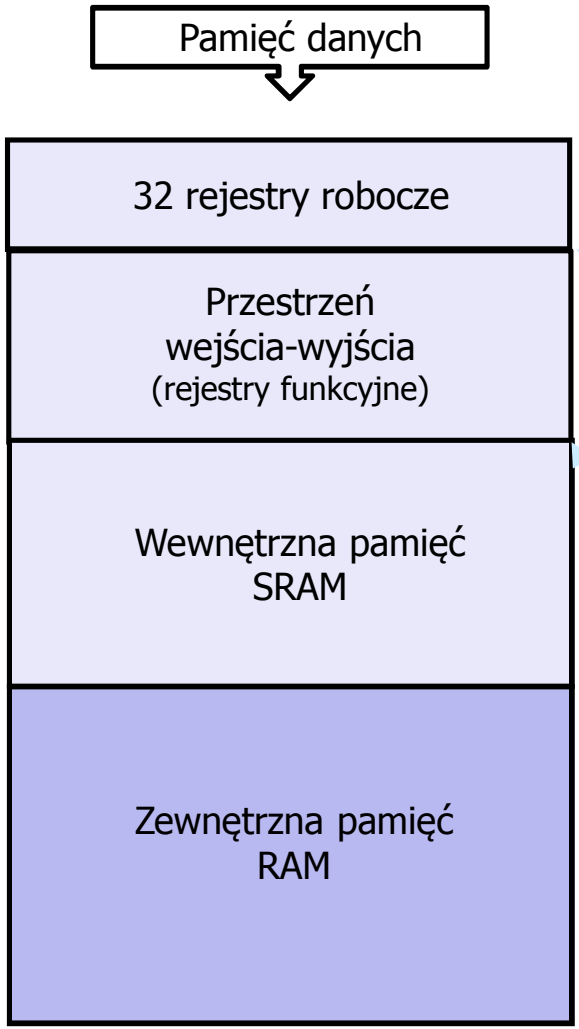
Nazwa dodatkowa

XL (LSB rejestru X)
XH (HSB rejestru X)
YL (LSB rejestru Y)
YH (HSB rejestru Y)
ZL (LSB rejestru Z)
ZH (HSB rejestru Z)

Rejestry X,Y,Z

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Przestrzeń wejścia-wyjścia (rejestry funkcyjne)



Rejestry funkcyjne

Zbiór rejestrów za pomocą których mikroprocesor komunikuje się ze światem zewnętrznym. Dodatkowo zawiera zbiór rejestrów za pomocą których konfigurujemy i kontrolujemy bloki funkcjonalne mikroprocesora.

adres	nazwa	funkcja
0x0000	TWBR	TWBR
0x0001	TWSR	TWSR
0x0002	TWAR	TWAR
0x0010	PIND	Rejestr linii wejściowych portu D
0x0011	DDRD	Rejestr określający kierunki poszczególnych bitów portu D
0x0012	PORTD	Rejestr danych portu D
0x0013	PINC	Rejestr linii wejściowych portu C
0x0014	DDRC	Rejestr określający kierunki poszczególnych bitów portu C
0x0015	PORTC	Rejestr danych portu C
0x0016	PINB	Rejestr linii wejściowych portu B
0x0017	DDRB	Rejestr określający kierunki poszczególnych bitów portu B
0x0018	PORTB	Rejestr danych portu B
0x0019	PINA	Rejestr linii wejściowych portu A
0x001A	DDRA	Rejestr określający kierunki poszczególnych bitów portu A
0x001B	PORTA	Rejestr danych portu A
0x003f	SREG	Rejestr statusów

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Przestrzeń wejścia-wyjścia (rejestry funkcyjne)

adres	nazwa	funkcja
0x0000	TWBR	TWBR
0x0001	TWSR	TWSR
0x0002	TWAR	TWAR
0x0010	PIND	Rejestr linii wejściowych portu D
0x0011	DDRD	Rejestr określający kierunki poszczególnych bitów portu D
0x0012	PORTD	Rejestr danych portu D
0x0013	PINC	Rejestr linii wejściowych portu C
0x0014	DDRC	Rejestr określający kierunki poszczególnych bitów portu C
0x0015	PORTC	Rejestr danych portu C
0x0016	PINB	Rejestr linii wejściowych portu B
0x0017	DDRB	Rejestr określający kierunki poszczególnych bitów portu B
0x0018	PORTB	Rejestr danych portu B
0x0019	PINA	Rejestr linii wejściowych portu A
0x001A	DDRA	Rejestr określający kierunki poszczególnych bitów portu A
0x001B	PORTA	Rejestr danych portu A
0x003f	SREG	Rejestr statusów

Przestrzeń wejścia-wyjścia (rejestry funkcyjne)

Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	10
$3E ($5E)	SPH	–	–	–	–	SP11	SP10	SP9	SP8	12
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	12
$3C ($5C)	OCR0	Timer/Counter0 Output Compare Register								82
$3B ($5B)	GICR	INT1	INT0	INT2	–	–	–	IVSEL	IVCE	47, 67
$3A ($5A)	GIFR	INTF1	INTF0	INTF2	–	–	–	–	–	68
$39 ($59)	TIMSK	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	82, 112, 130
$38 ($58)	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	83, 112, 130
$37 ($57)	SPMCR	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	248
$36 ($56)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	177
$35 ($55)	MCUCR	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	32, 66
$34 ($54)	MCUCSR	JTD	ISC2	–	JTRF	WDRF	BORF	EXTRF	PORF	40, 67, 228
$33 ($53)	TCCR0	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	80
$32 ($52)	TCNT0	Timer/Counter0 (8 Bits)								82
$31[1] ($51)[1]	OSCCAL	Oscillator Calibration Register								30
	OCDR	On-Chip Debug Register								224
$30 ($50)	SFIOR	ADTS2	ADTS1	ADTS0	–	ACME	PUD	PSR2	PSR10	56,85,131,198,218
$2F ($4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	107
$2E ($4E)	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	110
$2D ($4D)	TCNT1H	Timer/Counter1 – Counter Register High Byte								111
$2C ($4C)	TCNT1L	Timer/Counter1 – Counter Register Low Byte								111
$2B ($4B)	OCR1AH	Timer/Counter1 – Output Compare Register A High Byte								111
$2A ($4A)	OCR1AL	Timer/Counter1 – Output Compare Register A Low Byte								111
$29 ($49)	OCR1BH	Timer/Counter1 – Output Compare Register B High Byte								111
$28 ($48)	OCR1BL	Timer/Counter1 – Output Compare Register B Low Byte								111
$27 ($47)	ICR1H	Timer/Counter1 – Input Capture Register High Byte								111
$26 ($46)	ICR1L	Timer/Counter1 – Input Capture Register Low Byte								111
$25 ($45)	TCCR2	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	125
$24 ($44)	TCNT2	Timer/Counter2 (8 Bits)								127
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register								127
$22 ($42)	ASSR	–	–	–	–	AS2	TCN2UB	OCR2UB	TCR2UB	128
$21 ($41)	WDTCR	–	–	–	WDTOE	WDE	WDP2	WDP1	WDP0	42
$20[2] ($40)[2]	UBRRH	URSEL	–	–	–	UBRR[11:8]				164
	UCSRC	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	162

Przestrzeń wejścia-wyjścia (rejestry funkcyjne)

$1F ($3F)	EEARH	–	–	–	–	–	–	EEAR9	EEAR8	19
$1E ($3E)	EEARL	EEPROM Address Register Low Byte								19
$1D ($3D)	EEDR	EEPROM Data Register								19
$1C ($3C)	EECR	–	–	–	–	EERIE	EEMWE	EEWE	EERE	19
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	64
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	64
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	64
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	64
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	64
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	65
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	65
$14 ($34)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	65
$13 ($33)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	65
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	65
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	65
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	65
$0F ($2F)	SPDR	SPI Data Register								138
$0E ($2E)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X	138
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	136
$0C ($2C)	UDR	USART I/O Data Register								159
$0B ($2B)	UCSRA	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM	160
$0A ($2A)	UCSRB	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8	161
$09 ($29)	UBRRL	USART Baud Rate Register Low Byte								164
$08 ($28)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	199
$07 ($27)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	214
$06 ($26)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	216
$05 ($25)	ADCH	ADC Data Register High Byte								217
$04 ($24)	ADCL	ADC Data Register Low Byte								217
$03 ($23)	TWDR	Two-wire Serial Interface Data Register								179
$02 ($22)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	179

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$01 ($21)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	178
$00 ($20)	TWBR	Two-wire Serial Interface Bit Rate Register								177

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Przestrzeń wejścia-wyjścia (rejestry funkcyjne)

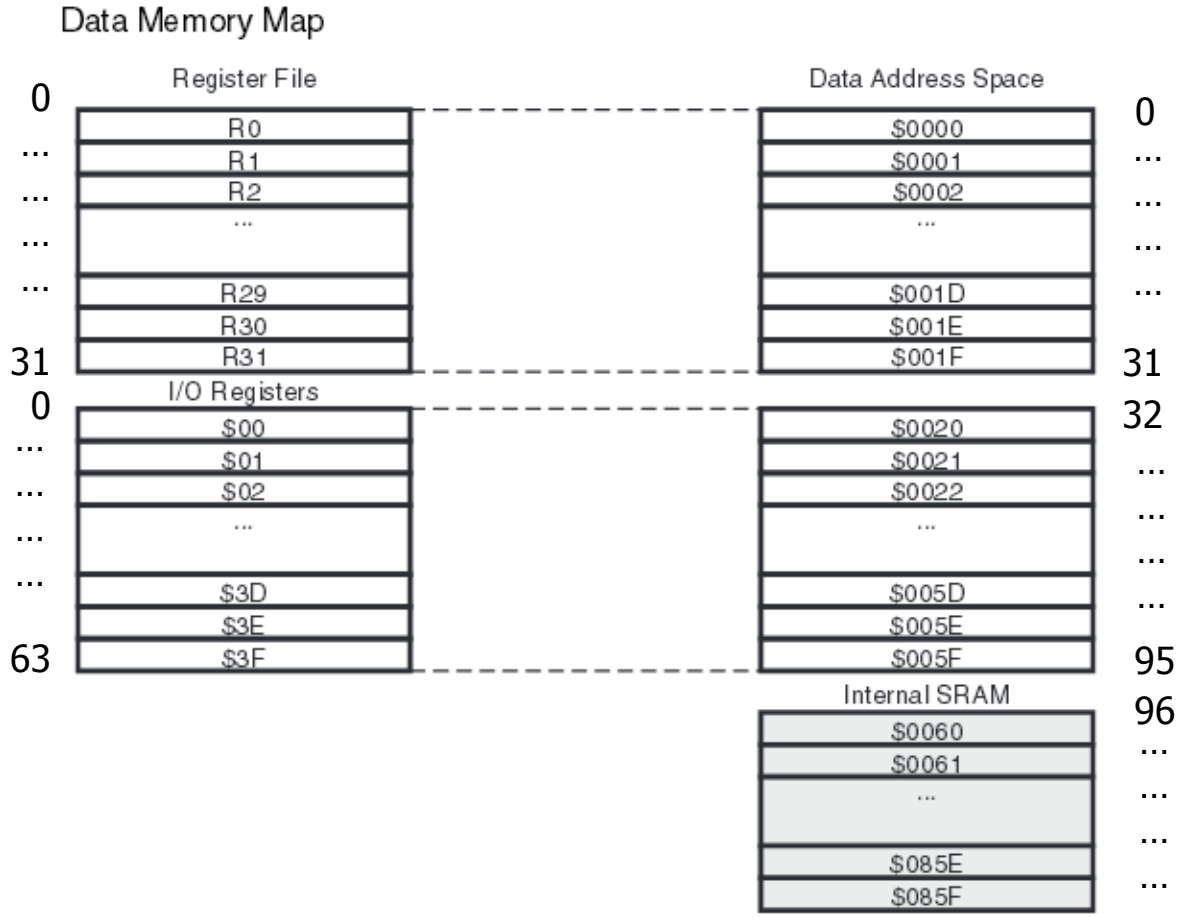
Przypisanie nazw adresom rejestrów funkcyjnych jest zdefiniowane w piku m32def.inc

```
.equ  SREG   = 0x3f
.equ  SPL    = 0x3d
.equ  SPH    = 0x3e
.equ  OCR0   = 0x3c
.equ  GICR   = 0x3b
.equ  GIFR   = 0x3a
.equ  TIMSK  = 0x39
.equ  TIFR   = 0x38
.equ  SPMCR  = 0x37
.equ  TWCR   = 0x36
.equ  MCUCR  = 0x35
.equ  MCUCSR = 0x34
.equ  TCCR0  = 0x33
.equ  TCNT0  = 0x32
.equ  OSCCAL = 0x31
.equ  OCDR   = 0x31
.equ  SFIOR  = 0x30
.equ  TCCR1A = 0x2f
.equ  TCCR1B = 0x2e
.equ  TCNT1L = 0x2c
.equ  TCNT1H = 0x2d
.equ  OCR1AL = 0x2a
.equ  OCR1AH = 0x2b
.equ  OCR1BL = 0x28
.equ  OCR1BH = 0x29
.equ  ICR1L  = 0x26
.equ  ICR1H  = 0x27
.equ  TCCR2  = 0x25
.equ  TCNT2  = 0x24
.equ  OCR2   = 0x23
.equ  ASSR   = 0x22
```

```
.equ  EEARH  = 0x1f
.equ  EEDR   = 0x1d
.equ  EECR   = 0x1c
.equ  PORTA  = 0x1b
.equ  DDRA   = 0x1a
.equ  PINA   = 0x19
.equ  PORTB  = 0x18
.equ  DDRB   = 0x17
.equ  PINB   = 0x16
.equ  PORTC  = 0x15
.equ  DDRC   = 0x14
.equ  PINC   = 0x13
.equ  PORTD  = 0x12
.equ  DDRD   = 0x11
.equ  PIND   = 0x10
.equ  SPDR   = 0x0f
.equ  SPSR   = 0x0e
.equ  SPCR   = 0x0d
.equ  UDR    = 0x0c
.equ  UCSRA  = 0x0b
.equ  UCSRB  = 0x0a
.equ  UBRRL  = 0x09
.equ  ACSR   = 0x08
.equ  ADMUX  = 0x07
.equ  ADCSRA = 0x06
.equ  ADCH   = 0x05
.equ  ADCL   = 0x04
.equ  TWDR   = 0x03
.equ  TWAR   = 0x02
.equ  TWSR   = 0x01
.equ  TWBR   = 0x00
```

Przestrzeń adresowa pamięci danych

Podsumowanie adresacji



Podstawowe Rejestry Funkcyjne

SPH:SPL
Wskaźnik Stosu (Stack Pointer)

STOS

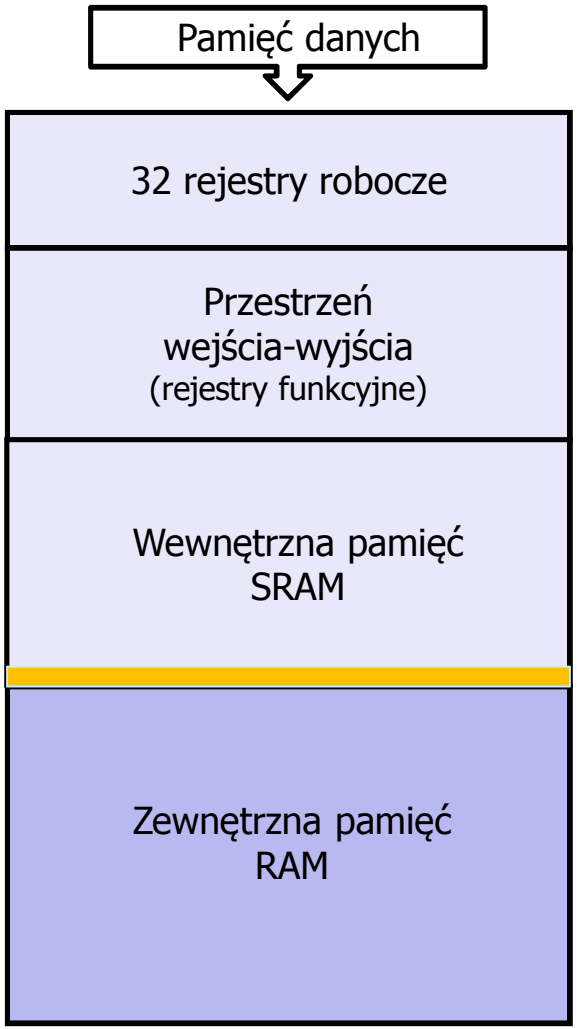
Stos jest obszarem pamięci wydzielonym z pamięci SRAM i służy do chwilowego przechowywania zawartości rejestrów.
Operacje zapisu (push) i odczytu (pop).
Obsługa przerwań i podprogramów.
Struktura LIFO.

Stos (ang. Stack) – liniowa struktura danych, w której dane dokładane są na wierzch stosu i z wierzchołka stosu są pobierane (bufor typu **LIFO**, *Last In, First Out*; *ostatni na wejściu, pierwszy na wyjściu*)

Przeciwieństwem stosu jest kolejka, bufor typu **FIFO** (ang. *First In, First Out*; *pierwszy na wejściu, pierwszy na wyjściu*), w którym dane obsługiwane są w takiej kolejności, w jakiej zostały dostarczone (jak w kolejce do kasy).

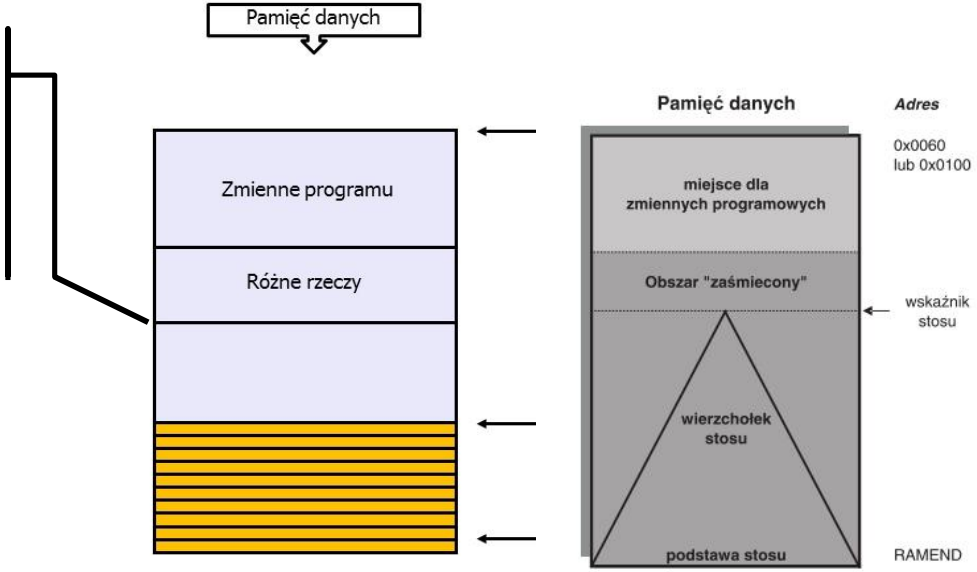
LIFO – Last In First Out
FIFO – First In First Out

Wewnętrzna pamięć SRAM

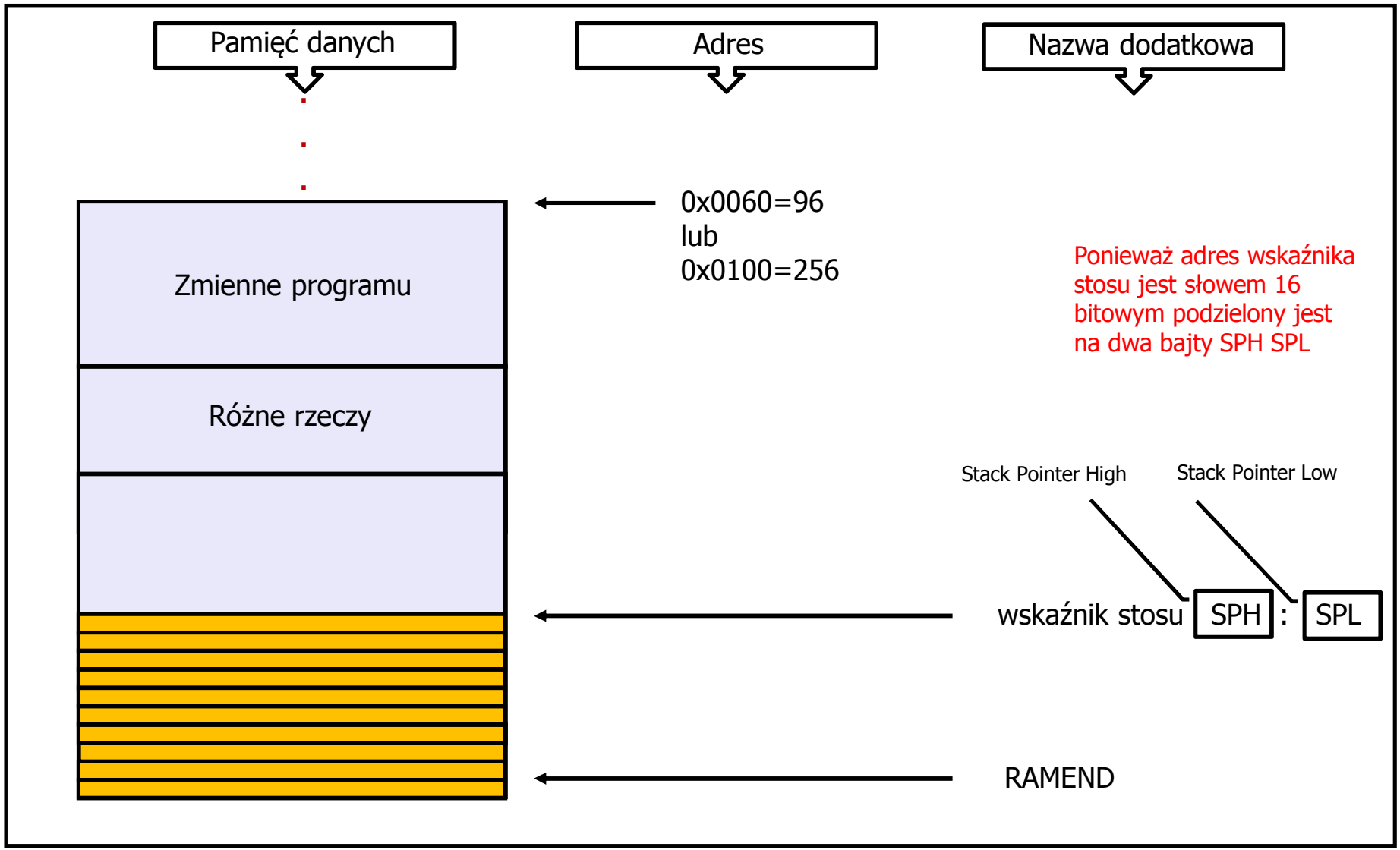


STOS

Stos jest wydzielony z pamięci SRAM i służy do chwilowego przechowywania zawartości rejestrów
Operacje zapisu (push) i odczytu (pop).
Obsługa przerwań i podprogramów.
Struktura LIFO.

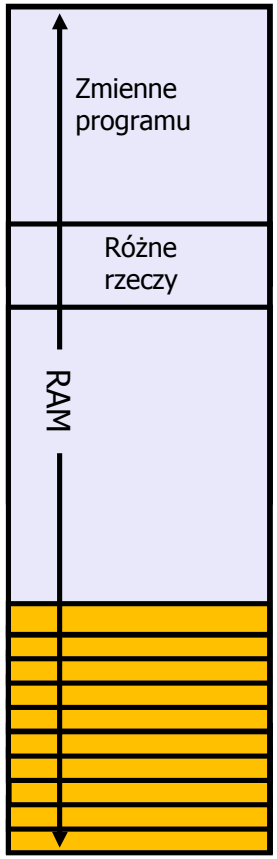


Wewnętrzna pamięć SRAM - STOS



Podstawowe Rejestry Funkcyjne

SPH:SPL
Wskaźnik Stosu (Stack Pointer)



Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

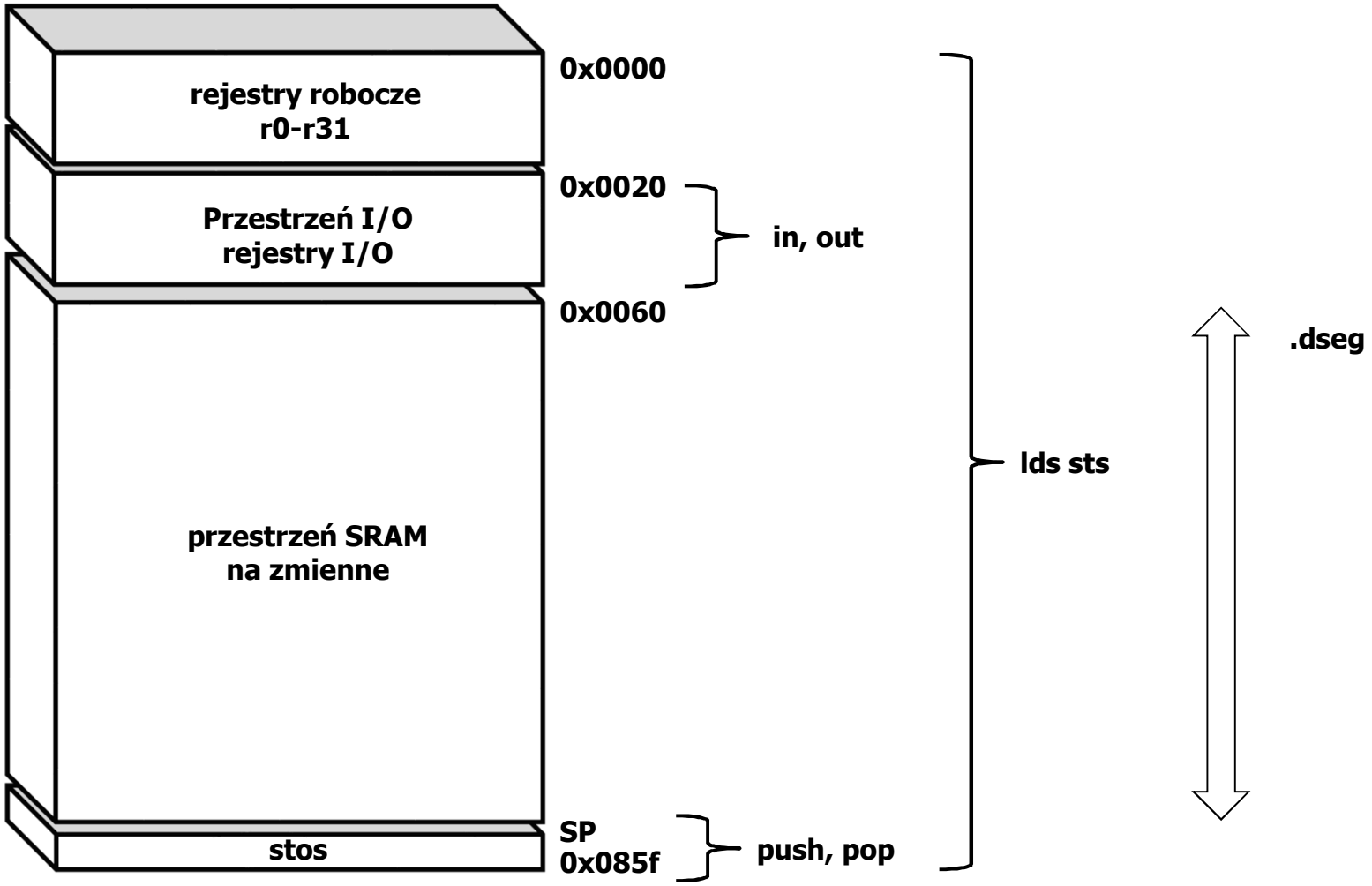
Wskaźnik Stosu (Stack Pointer) jest 16 bitowy

0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1

SPH **SPL**

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Przestrzeń danych - powtórzenie



WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Pierwszy program – środowisko AVR Studio

Program 1 - Inicjalizacja stosu

```
.include "m32def.inc"
.cseg
.org        0x0000

ldi         r16,high(ramend)
out         sph,r16
ldi         r16, low(ramend)
out         spl,r16

ldi         r16,0x55
push        r16
pop         r17

.exit
```

Włącza do kompilacji wskazany plik

Dalsza część zapisów dotyczy programu

Jeżeli dyrektywa .org znajduje się w .cseg to określa miejsce w pamięci FLASH gdzie kompilator zacznie umieszczać generowane bajty programu.

Instrukcje asemblera

Odkładanie i popieranie wartości 0x55 rejestrur16 na stosie

Kończy kompilację danego pliku

Pierwszy program – środowisko AVR Studio

```
.include "m32def.inc"
```

Włącza do kompilacji wskazany plik definiuje nam etykietę RAMEND czyli podaję ostatni adres pamięci RAM dla mikrokontrolera Atmega32

```
; ***** CPU REGISTER DEFINITIONS ************************************************
.def            XH              = r27
.def            XL              = r26
.def            YH              = r29
.def            YL              = r28
.def            ZH              = r31
.def            ZL              = r30



; ***** DATA MEMORY DECLARATIONS ************************************************
.equ            FLASHEND        = 0x3fff        ; Note: Word address
.equ            IOEND           = 0x003f
.equ            SRAM_START      = 0x0060
.equ            SRAM_SIZE       = 2048
.equ            RAMEND          = 0x085f
.equ            XRAMEND         = 0x0000
.equ            E2END           = 0x03ff
.equ            EEPROMEND       = 0x03ff
.equ            EEADRBITS       = 10
```

Program bez włączenia do kompilacji pliku m32def.inc nie będzie kompilowany. Kompilator nie będzie wiedział co to jest RAMEND

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Pierwszy program – środowisko AVR Studio

Symulacja programu inicjalizacji stosu