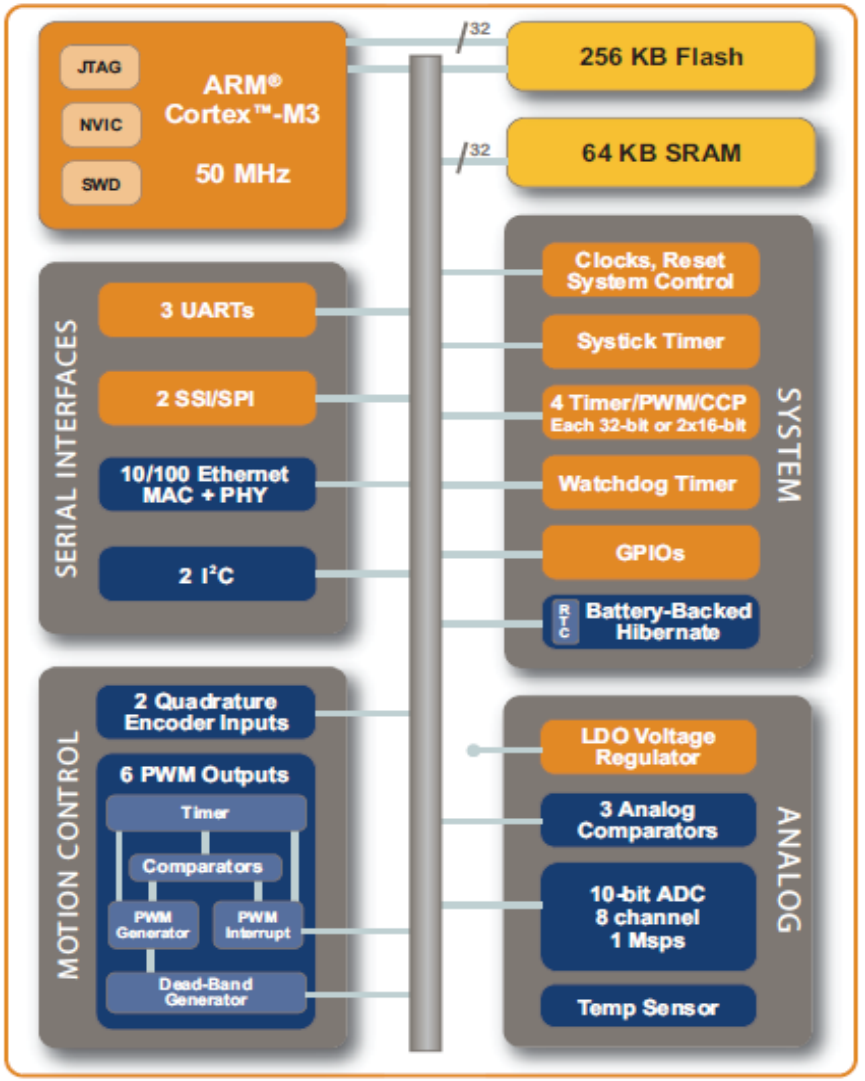
Wykład 3

Architektura rdzenia CortexM3

Architektura mikrokontrolera Stellaris



Cortex M3

- zestaw instrukcji Thumb-2, w pełni kompatybilny z Thumb®
- prędkość wykonywania operacji: 25 lub 50 MHz
- zintegrowany kontroler przerwań NVIC (Nested Vectored Interrupt Controller)
- jednotaktowe pamięci Flash / SRAM o pojemności: 64/16 kB (LM3S6100), 96/32 kB (LM3S6400), 128/32 (LM3S6600), 128/64 kB (LM3S6700), 256/64 kB (LM3S6900)
- 1 lub 2 interfejsy 10/100 Ethernet MAC / PHY
- 3 lub 4 układy czasowe, każdy konfigurowany jako jeden 32-bitowy lub dwa 16-bitowe z trybem PWM
- zegar czasu rzeczywistego (RTC)
- 32-bitowy Watchdog

Cortex M3

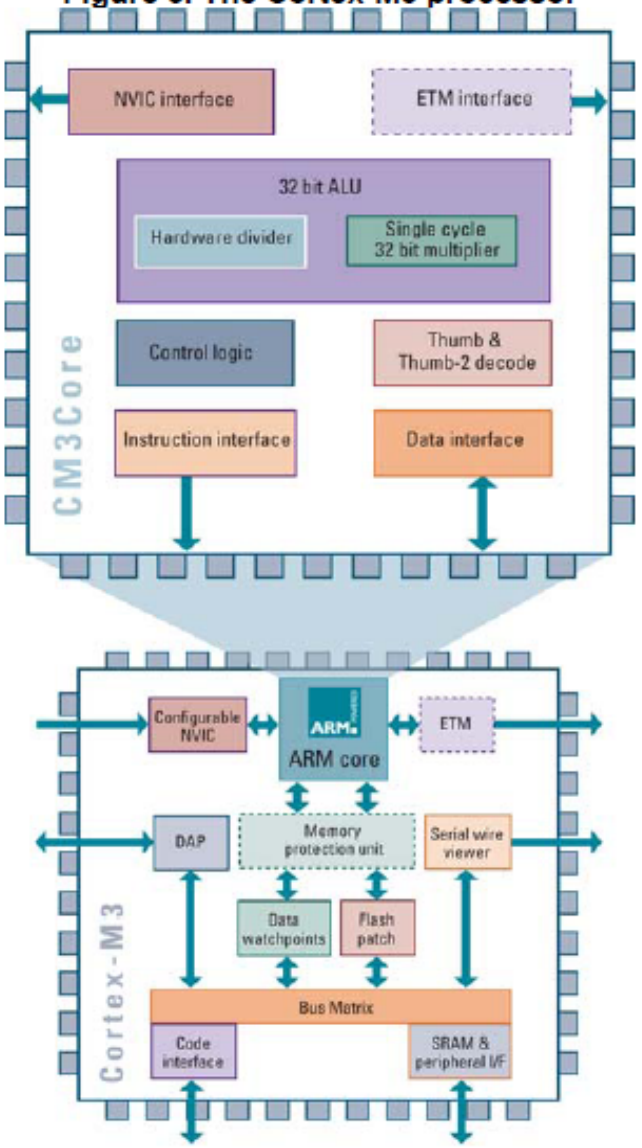
- 1 lub 2 synchroniczne interfejsy szeregowe (SSI) do realizacji interfejsów SPI, MICROWIRE lub TI
- interfejsy JTAG i Serial Wire dla debbugowania
- 1...3 programowalne układy UART typu 16C550
- 1...3 komparatory analogowe
- 2...8-kanałowy 10-bitowy przetwornik A/C
- do 6 wyjść PWM do sterowania mechanizmami
- do 2 enkoderów kwadraturowych (QEI) do sterowania ruchem
- do 2 interfejsów I2C
- do 46 portów I/O ogólnego zastosowania, zależnie od konfiguracji użytkownika

Cortex M3

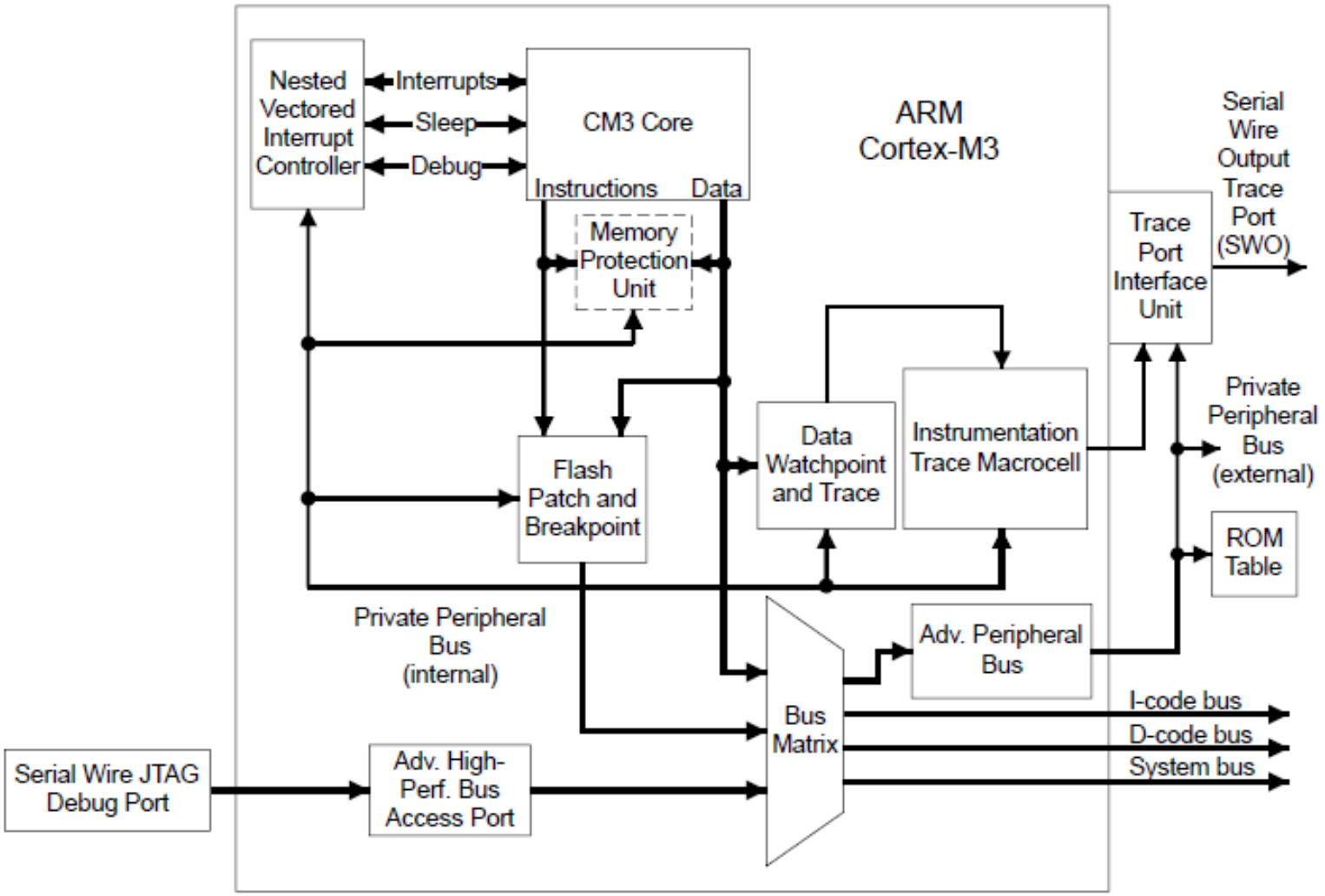
- precyzyjny skokowy stabilizator napięcia (LDO)
- zróżnicowane źródła zerowania mikrokontrolera: włączenie zasilania (POR), spadek napięcia zasilania (BOR), zerowanie programowe, układ Watchdog
- zróżnicowane tryby oszczędzania energii: dwa tryby uśpienia procesora, programowe wyłączanie poszczególnych układów peryferyjnych
- przemysłowy zakres temperatury pracy: -40...+85 °C
- obudowy 100-LQFP, zgodne ze standardem RoHS

Rdzeń CortexM3

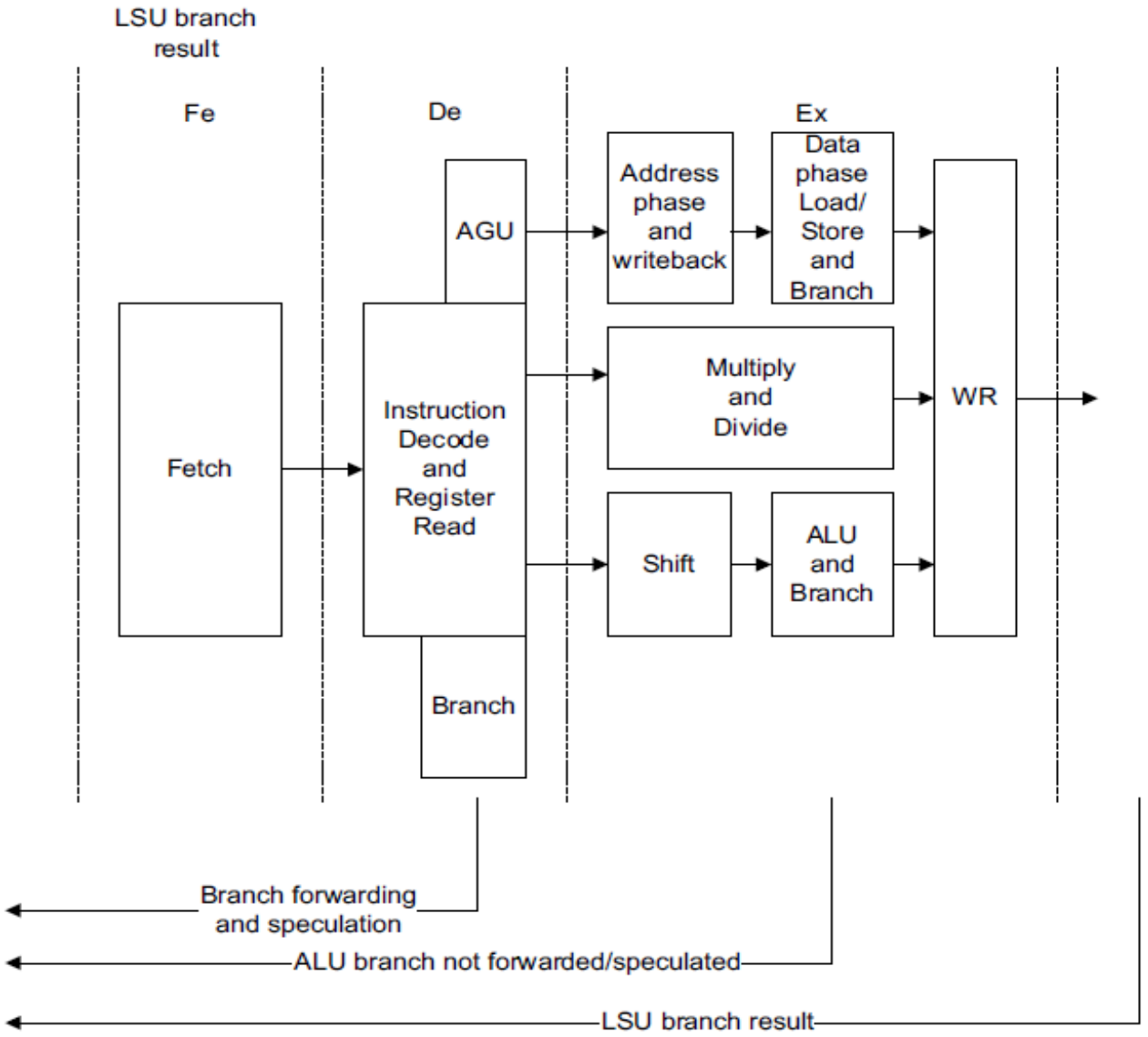
Figure 3. The Cortex-M3 processor



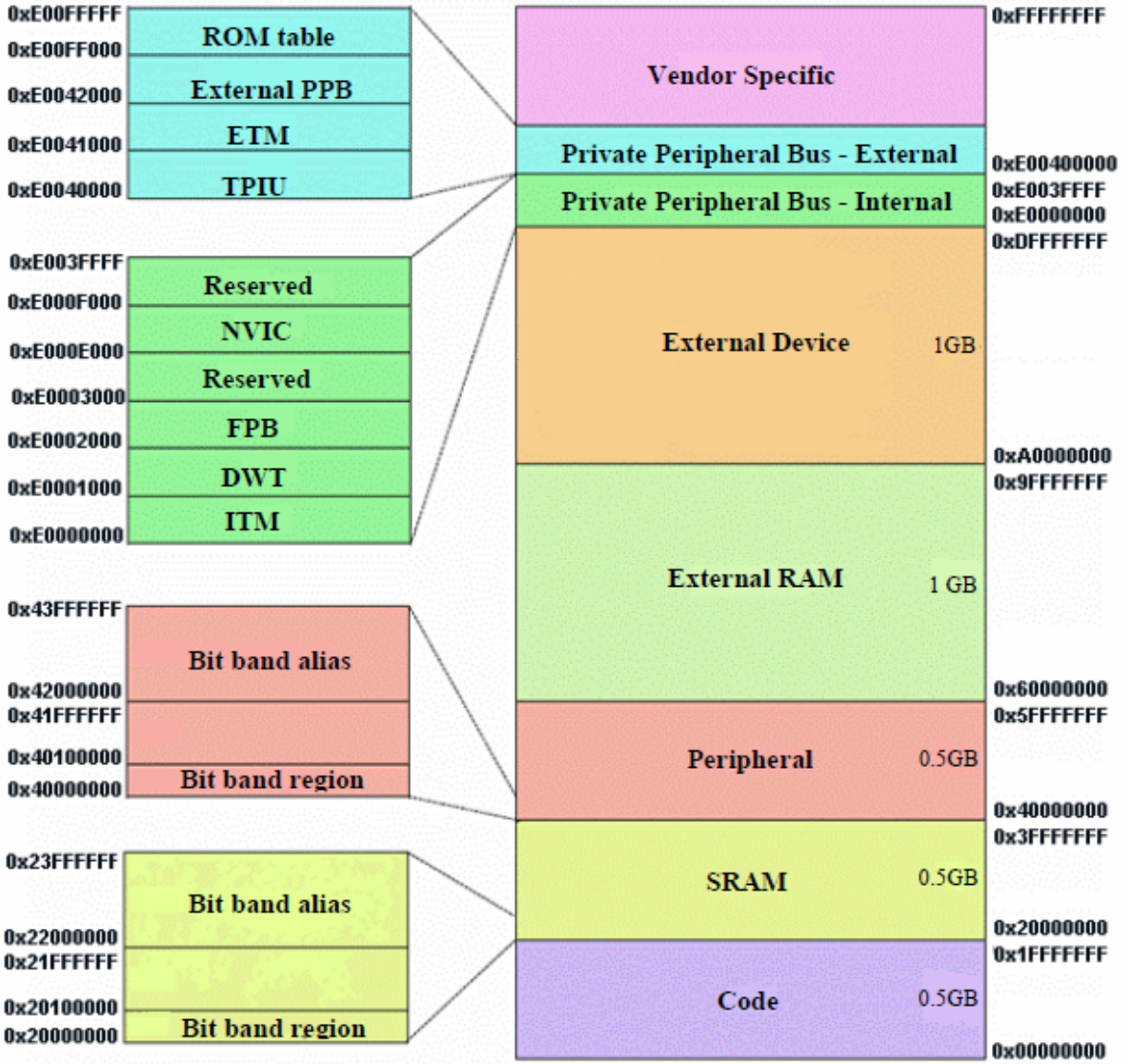
Schemat blokowy CPU



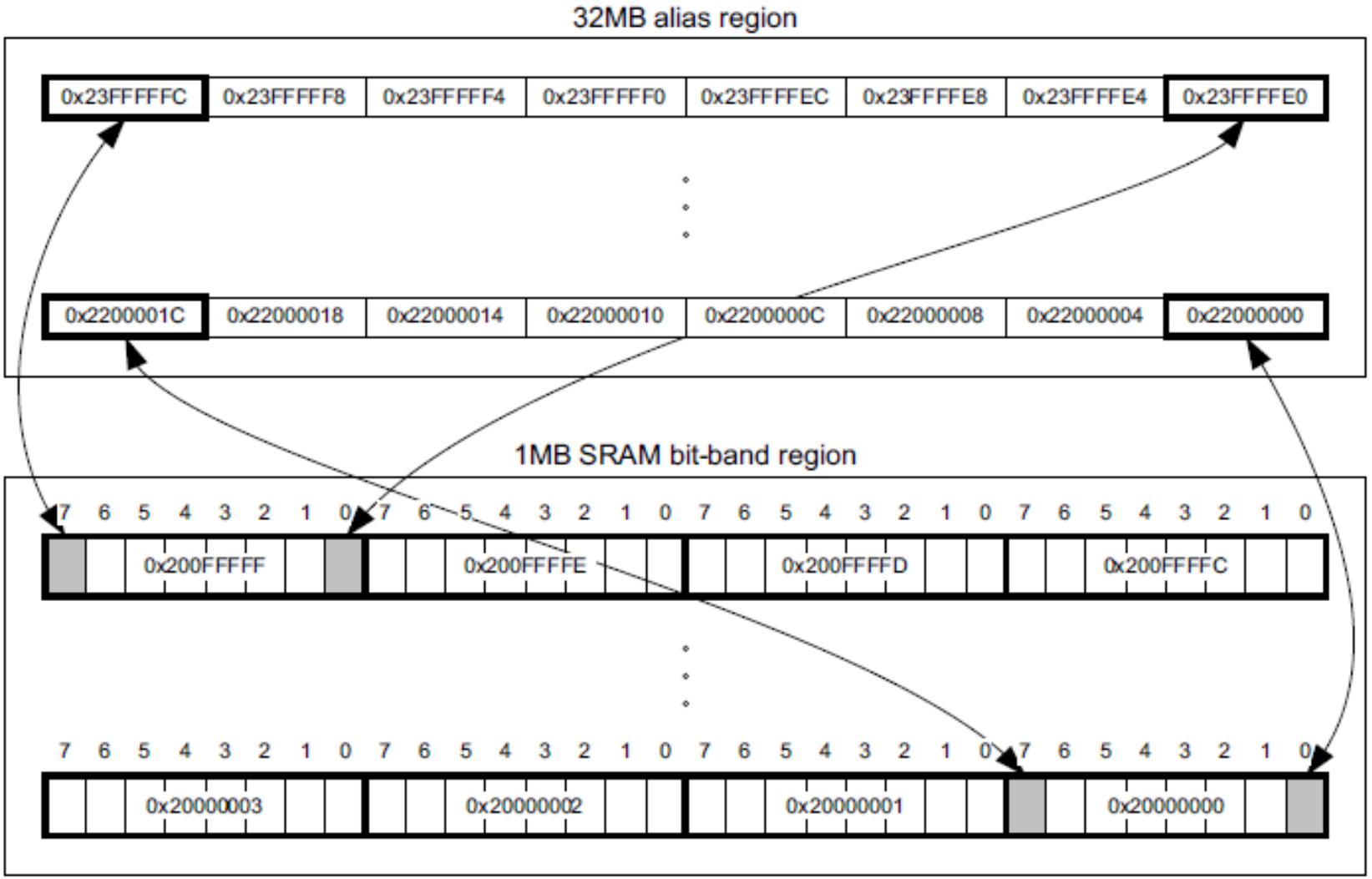
Potok wykonawczy



Mapa pamięci



Bit-band mapping



Mapa pamięci

Memory		
0x0000.0000	0x1FFF.FFFF	On-chip flash [b]
0x2000.0000	0x200F.FFFF	Bit-banded on-chip SRAM [c]
0x2010.0000	0x21FF.FFFF	Reserved non-bit-banded SRAM space
0x2200.0000	0x23FF.FFFF	Bit-band alias of 0x2000.0000 through 0x200F.FFFF
0x2400.0000	0x3FFF.FFFF	Reserved non-bit-banded SRAM space

Mapa pamięci

FiRM Peripherals		
0x4000.0000	0x4000.0FFF	Watchdog timer
0x4000.1000	0x4000.3FFF	Reserved
0x4000.4000	0x4000.4FFF	GPIO Port A
0x4000.5000	0x4000.5FFF	GPIO Port B
0x4000.6000	0x4000.6FFF	GPIO Port C
0x4000.7000	0x4000.7FFF	GPIO Port D
0x4000.8000	0x4000.8FFF	SSI0
0x4000.9000	0x4000.9FFF	SSI1
0x4000.A000	0x4000.BFFF	Reserved
0x4000.C000	0x4000.CFFF	UART0
0x4000.D000	0x4000.DFFF	UART1
0x4000.E000	0x4000.EFFF	UART2
0x4000.F000	0x4000.FFFF	Reserved
0x4001.0000	0x4001.FFFF	Reserved for future FiRM peripherals

Mapa pamięci

Peripherals		
0x4002.0000	0x4002.07FF	I2C Master 0
0x4002.0800	0x4002.0FFF	I2C Slave 0
0x4002.1000	0x4002.17FF	I2C Master 1
0x4001.1800	0x4002.1FFF	I2C Slave 1
0x4002.2000	0x4002.3FFF	Reserved
0x4002.4000	0x4002.4FFF	GPIO Port E
0x4002.5000	0x4002.5FFF	GPIO Port F
0x4002.6000	0x4002.6FFF	GPIO Port G
0x4002.7000	0x4002.7FFF	GPIO Port H
0x4002.8000	0x4002.8FFF	PWM
0x4002.9000	0x4002.BFFF	Reserved
0x4002.C000	0x4002.CFFF	QEI0
0x4002.D000	0x4002.DFFF	QEI1
0x4002.E000	0x4002.FFFF	Reserved
0x4003.0000	0x4003.0FFF	Timer0
0x4003.1000	0x4003.1FFF	Timer1
0x4003.2000	0x4003.2FFF	Timer2
0x4003.3000	0x4003.3FFF	Timer3

Mapa pamięci

0x4003.4000	0x4003.7FFF	Reserved
0x4003.8000	0x4003.8FFF	ADC
0x4003.9000	0x4003.BFFF	Reserved
0x4003.C000	0x4003.CFFF	Analog Comparators
0x4003.D000	0x4003.FFFF	Reserved
0x4004.0000	0x4004.0FFF	CAN0 Controller
0x4004.1000	0x4004.1FFF	CAN1 Controller
0x4004.3000	0x4004.7FFF	Reserved
0x4004.9000	0x4004.BFFF	Reserved
0x4004.C000	0x400F.BFFF	Reserved
0x400F.C000	0x400F.CFFF	Hibernation Module
0x400F.D000	0x400F.DFFF	Flash control
0x400F.E000	0x400F.EFFF	System control
0x400F.F000	0x400F.FFFF	Reserved
0x4011.1000	0x4011.1FFF	Reserved
0x4012.0000	0x41FF.FFFF	Reserved for non bit-banded peripheral space
0x4200.0000	0x43FF.FFFF	Bit-banded alias of 0x4000.0000 through 0x400F.FFFF
0x4400.0000	0x5E32.FFFF	Reserved for non bit-banded peripheral space
0x5E34.0000	0x5FFF.FFFF	Reserved
0x6000.0000	0xDFFF.FFFF	Reserved for external devices

Mapa pamięci

Private Peripheral Bus		
0xE000.0000	0xE000.0FFF	Instrumentation Trace Macrocell (ITM)
0xE000.1000	0xE000.1FFF	Data Watchpoint and Trace (DWT)
0xE000.2000	0xE000.2FFF	Flash Patch and Breakpoint (FPB)
0xE000.3000	0xE000.DFFF	Reserved
0xE000.E000	0xE000.EFFF	Nested Vectored Interrupt Controller (NVIC)
0xE000.F000	0xE003.FFFF	Reserved
0xE004.0000	0xE004.0FFF	Trace Port Interface Unit (TPIU)
0xE004.1000	0xE004.1FFF	Reserved
0xE004.2000	0xE00F.FFFF	Reserved
0xE010.0000	0xFFFF.FFFF	Reserved for vendor peripherals

W przypadku dostępu do obszarów zarezerwowanych zgłaszany jest błąd

Tryby pracy procesora

The processor supports two modes of operation, Thread mode and Handler mode:

- Thread mode is entered on Reset, and can be entered as a result of an exception return. Privileged and User (Unprivileged) code can run in Thread mode.
- Handler mode is entered as a result of an exception. All code is privileged in Handler mode.

The processor can operate in one of two operating states:

- Thumb state. This is normal execution running 16-bit and 32-bit halfword aligned Thumb and Thumb-2 instructions.
- Debug State. This is the state when in halting debug.

Thread mode is privileged out of reset, but you can change it to user or unprivileged by clearing the CONTROL[0] bit using the MSR instruction. User access prevents:

- use of some instructions such as CPS to set FAULTMASK and PRIMASK
- access to most registers in *System Control Space* (SCS).

Typy danych

The processor supports the following data types:

- 32-bit words
- 16-bit halfwords
- 8-bit bytes.

Typy danych

The processor supports the following data types:

- 32-bit words
- 16-bit halfwords
- 8-bit bytes.

Format pamięci

The processor views memory as a linear collection of bytes numbered in ascending order from 0. For example:

- bytes 0-3 hold the first stored word
- bytes 4-7 hold the second stored word.

The processor can access data words in memory in little-endian format or big-endian format. It always accesses code in little-endian format.

Note

Little-endian is the default memory format for ARM processors.

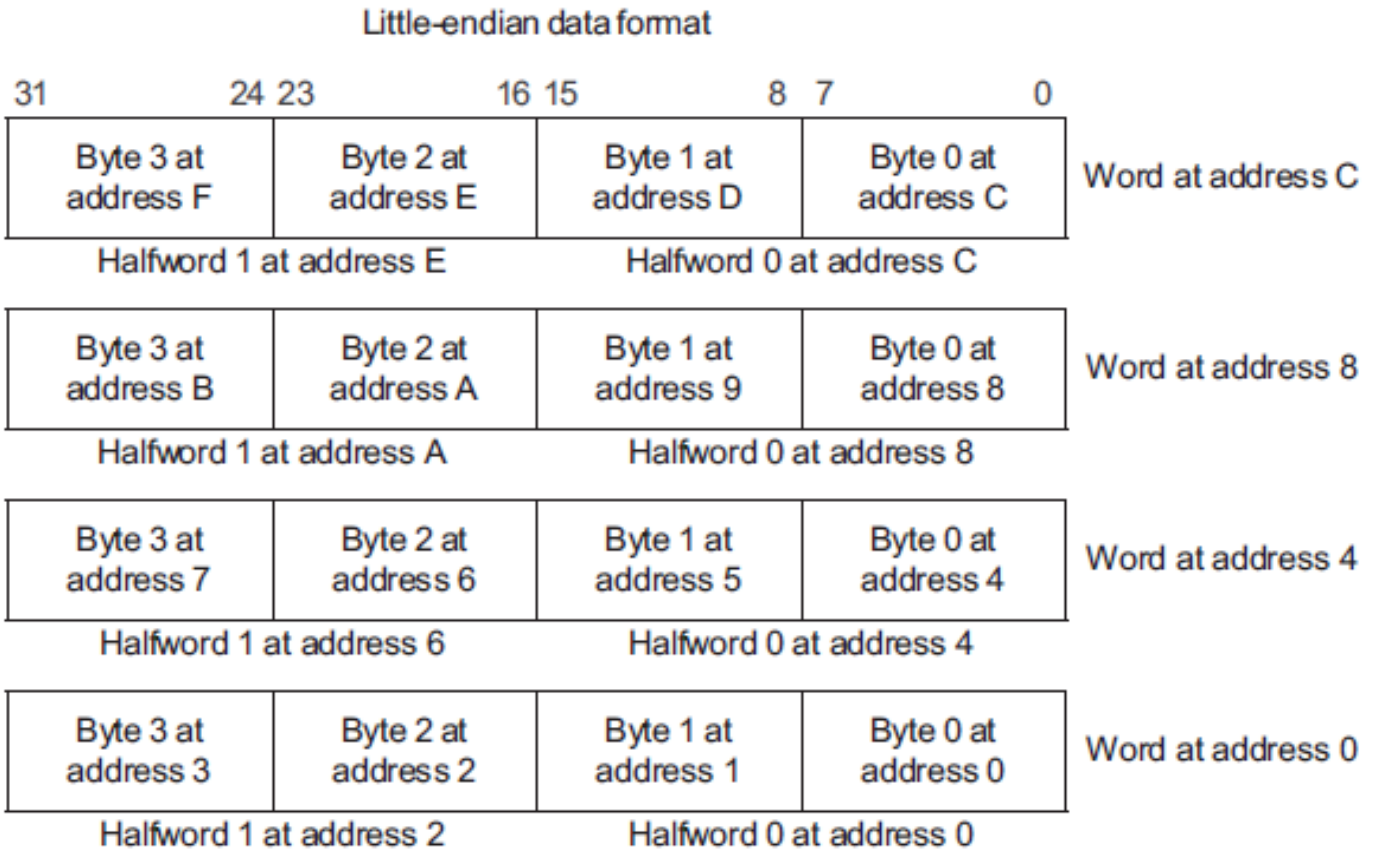
Format pamięci

The processor contains a configuration pin, **BIGEND**, that enables you to select either the little-endian or BE-8 big-endian format. This configuration pin is sampled on reset. You cannot change endianness when out of reset.

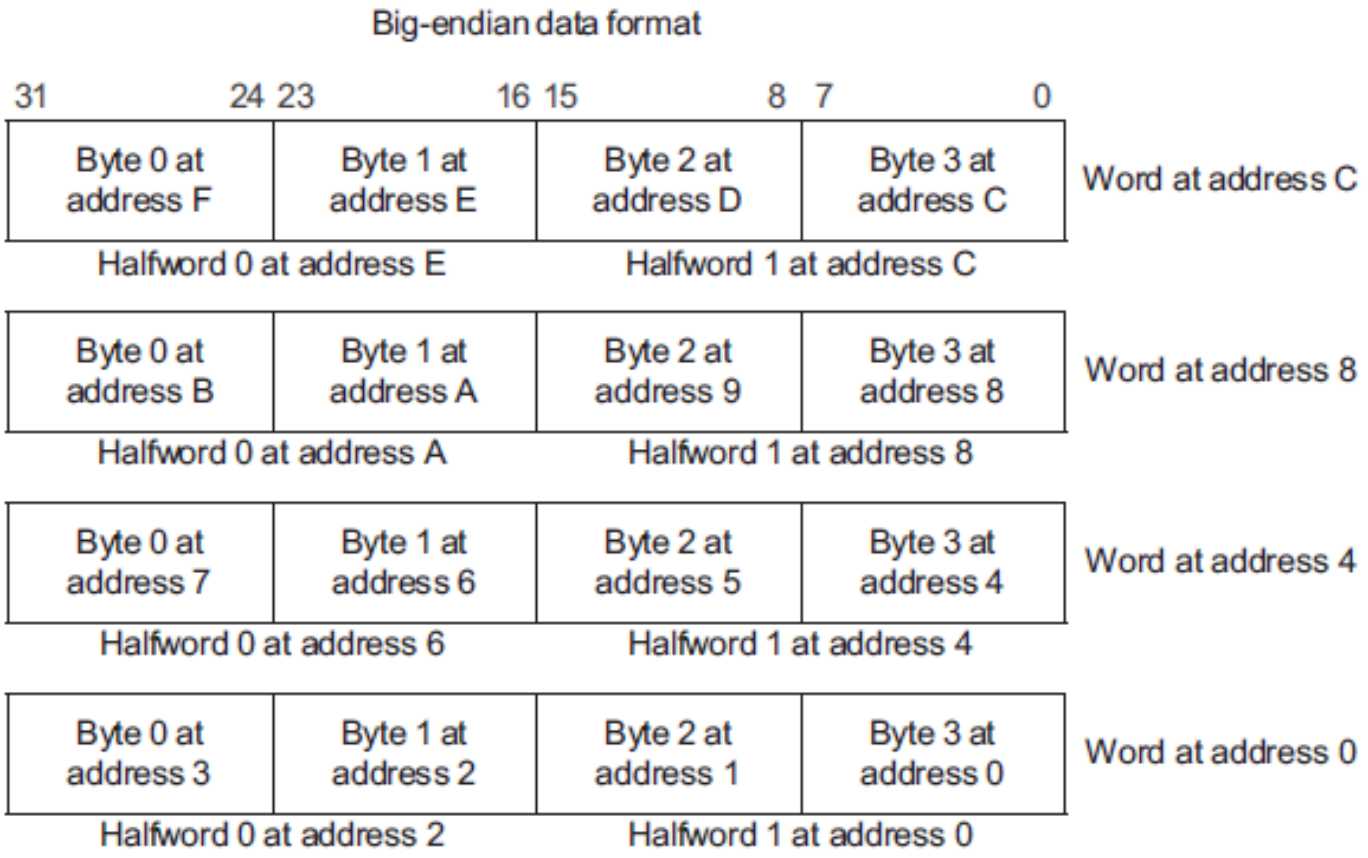
Note

- Accesses to *System Control Space* (SCS) are always little endian.
- Attempts to change endianness while not in reset are ignored.
- *Private Peripheral Bus* (PPB) space is little-endian, irrespective of the setting of **BIGEND**.

Format pamięci



Format pamięci

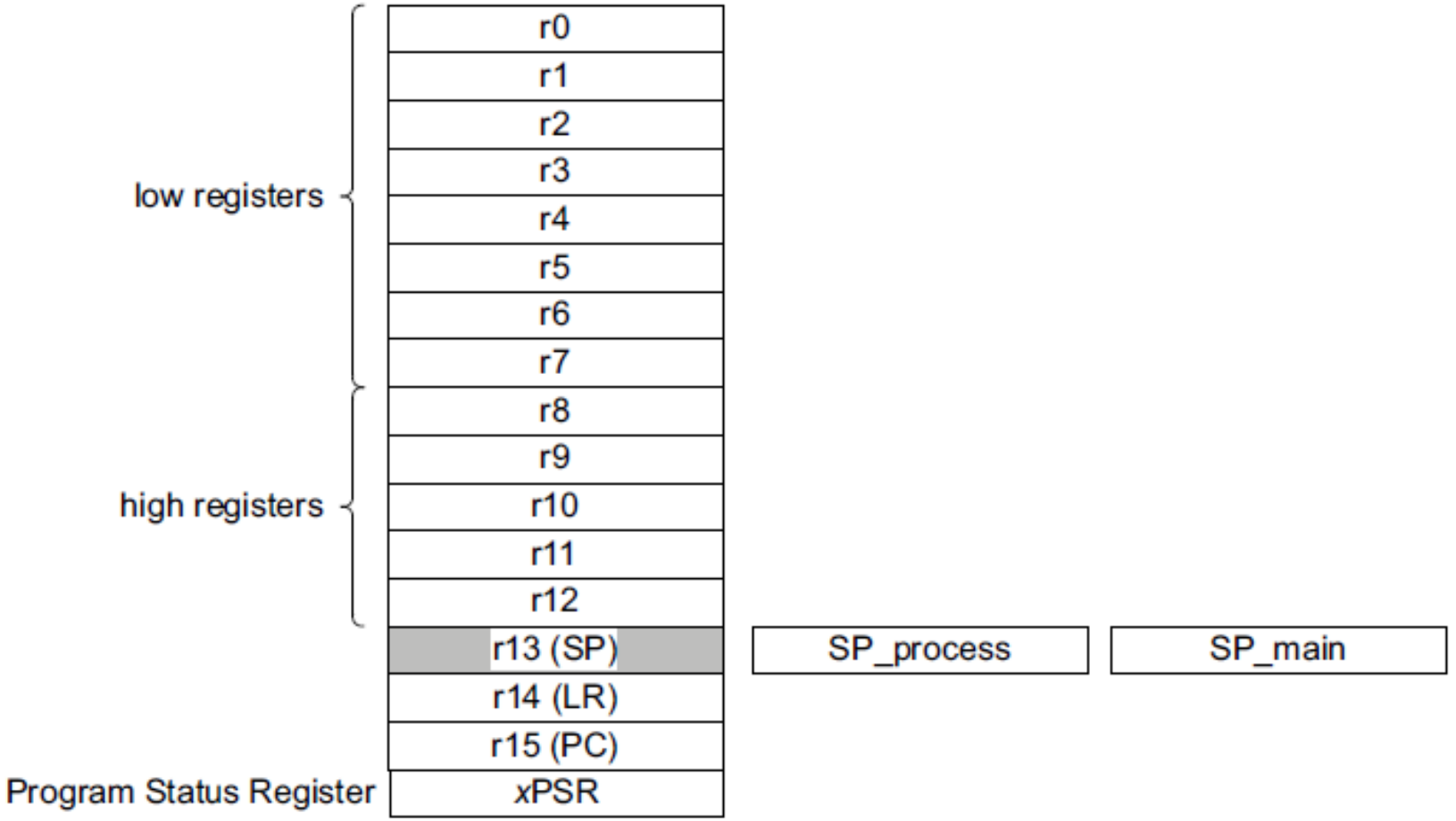


Rejestry procesora

Procesor Cortex M3 posiada następujące rejestry 32 bitowe:

- 13 rejestrów ogólnego przeznaczenia, r0-r12
- rejestr r13 używany jako wskaźnik stosu, mapowany w zależności od trybu pracy jako SP_process lub SP_main
- link register r14
- licznik programu (program counter) r15
- rejestr statusy xPSR

Rejestry procesora CortexM3



Rejestry procesora ARM7TDMI

System and User	FIQ	Supervisor	Abort	IRQ	Undefined
r0	r0	r0	r0	r0	r0
r1	r1	r1	r1	r1	r1
r2	r2	r2	r2	r2	r2
r3	r3	r3	r3	r3	r3
r4	r4	r4	r4	r4	r4
r5	r5	r5	r5	r5	r5
r6	r6	r6	r6	r6	r6
r7	r7	r7	r7	r7	r7
r8	r8_fiq	r8	r8	r8	r8
r9	r9_fiq	r9	r9	r9	r9
r10	r10_fiq	r10	r10	r10	r10
r11	r11_fiq	r11	r11	r11	r11
r12	r12_fiq	r12	r12	r12	r12
r13	r13_fiq	r13_svc	r13_abt	r13_irq	r13_und
r14	r14_fiq	r14_svc	r14_abt	r14_irq	r14_und
r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)

ARM state program status registers

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_fiq	SPSR_svc	SPSR_abt	SPSR_irq	SPSR_und

Rejestry universalne

The general-purpose registers r0-r12 have no special architecturally-defined uses. Most instructions that can specify a general-purpose register can specify r0-r12.

Low registers	Registers r0-r7 are accessible by all instructions that specify a general-purpose register.
High registers	Registers r8-r12 are accessible by all 32-bit instructions that specify a general-purpose register. Registers r8-r12 are not accessible by all 16-bit instructions.

Rejestry specjalne

Stack pointer	Register r13 is used as the *Stack Pointer* (SP). Because the SP ignores writes to bits [1:0], it is autoaligned to a word, four-byte boundary. Handler mode always uses SP_main, but you can configure Thread mode to use either SP_main or SP_process.
Link register	Register r14 is the subroutine *Link Register* (LR). The LR receives the return address from PC when a *Branch and Link* (BL) or *Branch and Link with Exchange* (BLX) instruction is executed. The LR is also used for exception return. At all other times, you can treat r14 as a general-purpose register.
Program counter	Register r15 is the *Program Counter* (PC). Bit [0] is always 0, so instructions are always aligned to word or halfword boundaries.

Rejestr APSR

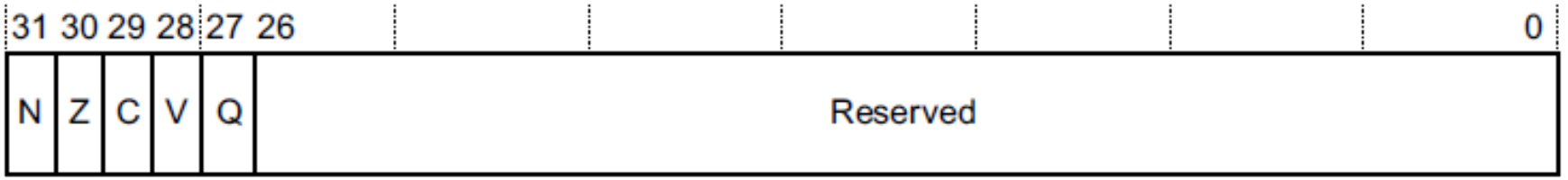


Figure 2-2 Application Program Status Register bit assignments

Rejestr APSR

Field	Name	Definition
[31]	N	Negative or less than flag: 1 = result negative or less than 0 = result positive or greater than.
[30]	Z	Zero flag: 1 = result of 0 0 = nonzero result.
[29]	C	Carry/borrow flag: 1 = carry or borrow 0 = no carry or borrow.
[28]	V	Overflow flag: 1 = overflow 0 = no overflow.
[27]	Q	Sticky saturation flag.
[26:0]	-	Reserved.

Rejestr IPSR

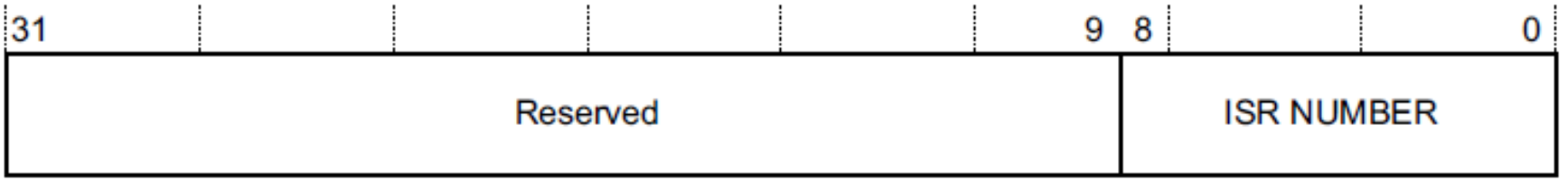


Figure 2-3 Interrupt Program Status Register bit assignments

Rejestr IPSR

Field	Name	Definition
[31:9]	-	Reserved.
[8:0]	ISR NUMBER	Number of pre-empted exception. Base level = 0 NMI = 2 SVCall = 11 INTISR[0] = 16 INTISR[1] = 17 . . . INTISR[15] = 31 . . . INTISR[239] = 255

Rejestr EPSR

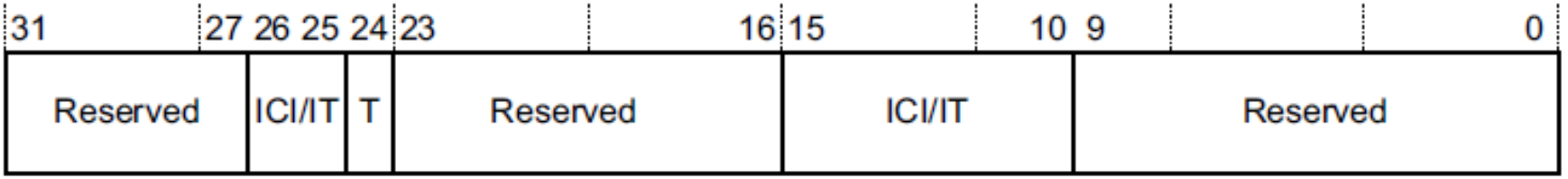


Figure 2-4 Execution Program Status Register

Rejestr EPSR

The *Execution PSR* (EPSR) contains two overlapping fields:

- the *Interruptible-Continuable Instruction* (ICI) field for interrupted load multiple and store multiple instructions
- the execution state field for the *If-Then* (IT) instruction, and the *Thumb state bit* (T-bit).

The EPSR is not directly accessible. Two events can modify the EPSR:

- an interrupt occurring during an LDM or STM instruction
- execution of the If-Then instruction.

Lista instrukcji

Instruction type	Size	Instructions
Data operations	16	ADC, ADD, AND, ASR, BIC, CMN, CMP, CPY, EOR, LSL, LSR, MOV, MUL, MVN, NEG, ORR, ROR, SBC, SUB, TST, REV, REVH, REVSH, SXTB, SXTH, UXTB, and UXTH.
Branches	16	B<cond>, B, BL, BX, and BLX. Note, no BLX with immediate.
Load-store single	16	LDR, LDRB, LDRH, LDRSB, LDRSH, STR, STRB, STRH.
Load-store multiple	16	LDMIA, POP, PUSH, and STMIA.
Exception generating	16	BKPT stops in debug if debug enabled, fault if debug disabled. SVC faults to the SVCall handler.
Data operations with immediate	32	ADC{S}. ADD{S}, CMN, RSB{S}, SBC{S}, SUB{S}, CMP, AND{S}, TST, BIC{S}, EOR{S}, TEQ, ORR{S}, MOV{S}, ORN{S}, and MVN{S}.
Data operations with large immediate	32	MOVW, MOVT, ADDW, and SUBW. MOVW and MOVT have a 16-bit immediate. This means they can replace literal loads from memory. ADDW and SUBW have a 12-bit immediate. This means they can replace many from memory literal loads.
Bit-field operations	32	BFI, BFC, UBFX, and SBFX. These are bitwise operations enabling control of position and size in bits. These both support C/C++ bit fields, in structs, in addition to many compare and some AND/OR assignment expressions.

Lista instrukcji

Instruction type	Size	Instructions
Data operations with three registers	32	ADC{S}. ADD{S}, CMN, RSB{S}, SBC{S}, SUB{S}, CMP, AND{S}, TST, BIC{S}, EOR{S}, TEQ, ORR{S}, MOV{S}, ORN{S}, and MVN{S}. No PKxxx instructions.
Shift operations	32	ASR{S}, LSL{S}, LSR{S}, RRX {S}, and ROR {S}.
Miscellaneous	32	REV, REVH, REVSH, RBIT, CLZ, SXTB, SXTH, UXTB, and UXTH. Extension instructions same as corresponding v6 16-bit instructions.
Table branch	32	TBB and TBH table branches for switch/case use. These are LDR with shifts and then branch.
Multiply	32	MUL, MLA, and MLS.
Multiply with 64-bit result	32	UMULL, SMULL, UMLAL, and SMLAL.

Lista instrukcji

Instruction type	Size	Instructions
Load-store addressing	32	Supports Format PC+/-imm12, Rbase+imm12, Rbase+/-imm8, and adjusted register including shifts. T variants used when in Privilege mode.
Load-store single	32	LDR, LDRB, LDRSB, LDRH, LDRSH, STR, STRB, STRH, and T variants. PLD and PLI are both hints and so act as a NOP.
Load-store multiple	32	STM, LDM, LDRD, and STRD.
Load-store exclusive	32	LDREX, STREX, LDREXB, LDREXH, STREXB, STREXH, CLREX. Fault if no local monitor. This is IMP DEF. LDREXD and STREXD are not included in this profile.
Branches	32	B, BL, and B<cond>. No BLX (1) because always changes state. No BXJ.

Lista instrukcji

Instruction type	Size	Instructions
System	32	MSR(2) and MRS(2) replace MSR/MRS but also do more. These access the other stacks and also the status registers. CPSIE/CPSID 32-bit forms are not supported. No RFE or SRS.
System	16	CPSIE and CPSID are quick versions of MSR(2) instructions and use the standard Thumb-2 encodings, but only permit use of i and f and not a.
Extended32	32	NOP (all forms), Coprocessor (MCR, MCR2, MCRR, MRC, MRC2, and MRRC), and YIELD (hinted NOP). Note, no MRS(1), MSR(1), or SUBS (PC return link).
Combined branch	16	CBZ and CBNZ (Compare and Branch if register is Zero or Non-Zero).
Extended	16	IT and NOP. This includes YIELD.
Divide	32	SDIV and UDIV. 32/32 divides both signed and unsigned with 32-bit quotient result, no remainder, it can be derived by subtraction. Early out is permitted.

Lista instrukcji

Instruction type	Size	Instructions
Sleep	16, 32	WFI, WFE, and SEV are in the class of hinted NOP instructions that control sleep behavior.
Barriers	32	ISB, DSB, and DMB are barrier instructions that ensure certain actions have taken place before the next instruction is executed.
Saturation	32	SSAT and USAT perform saturation on a register. They perform the following: Normalize the value using shift test for overflow from a selected bit position, the Q value. Set the xPSR Q bit if so, saturate the value if overflow detected. Saturation refers to the largest unsigned value or the largest/smallest signed value for the size selected.

Note

All coprocessor instructions generate a *NO CoProcessor* (NOCP) fault.