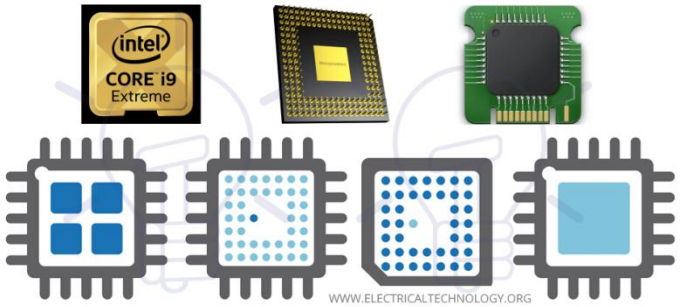


MİKROİŞLEMCİLER

Dr. Meltem KURT PEHLİVANOĞLU

W-7

MİKROİŞLEMCİLER

Digital Logic +

Digital Design +

Computer Architecture +

Microprocessors +

Microcontrollers +

Assembly Language Programming

MSP430

- 16-bit mikrodenetleyici ailesidir
- Düşük güç tüketim modları
- RISC mimarisi
- Kullanımı kolay
- Düşük maliyetli
- Ultra düşük güç tüketimi
- Gömülü ve sensör uygulamaları
- Endüstriyel Uygulamalar
- Alternatif Enerji Uygulamaları
- Tıbbi Uygulamalar
- Güvenlik Uygulamaları
- Akıllı Sayaç Uygulamaları
- Taşınabilir Tıbbi Cihazlar
- Taşınabilir Akıllı Aygıtlar

MSP430

- MSP430 bir mikrodenetleyici ailesidir ve çeşitli özelliklerdeki birçok mikrodenetleyiciye sahiptir.

- Texas Instruments ürün isimlendirme politikasına göre konuşursak MSP430F ve MSP430G (F serisinin biraz özellik yoksunu) şeklinde başlayan denetleyiciler flash memory yapıya sahip olmakla beraber MSP430FR ile başlayanlar FRAM'e yapıya sahiptirler

- FRAM: Non-volatile (yani geçici/uçucu olmayan) bir hafızadır, okuma/yazma hızı flash yapıya göre çok hızlıdır. Okuma/yazma ömrü (r/w cycle) çok fazladır (yaklaşık 100 trilyon kadar).

MSP430

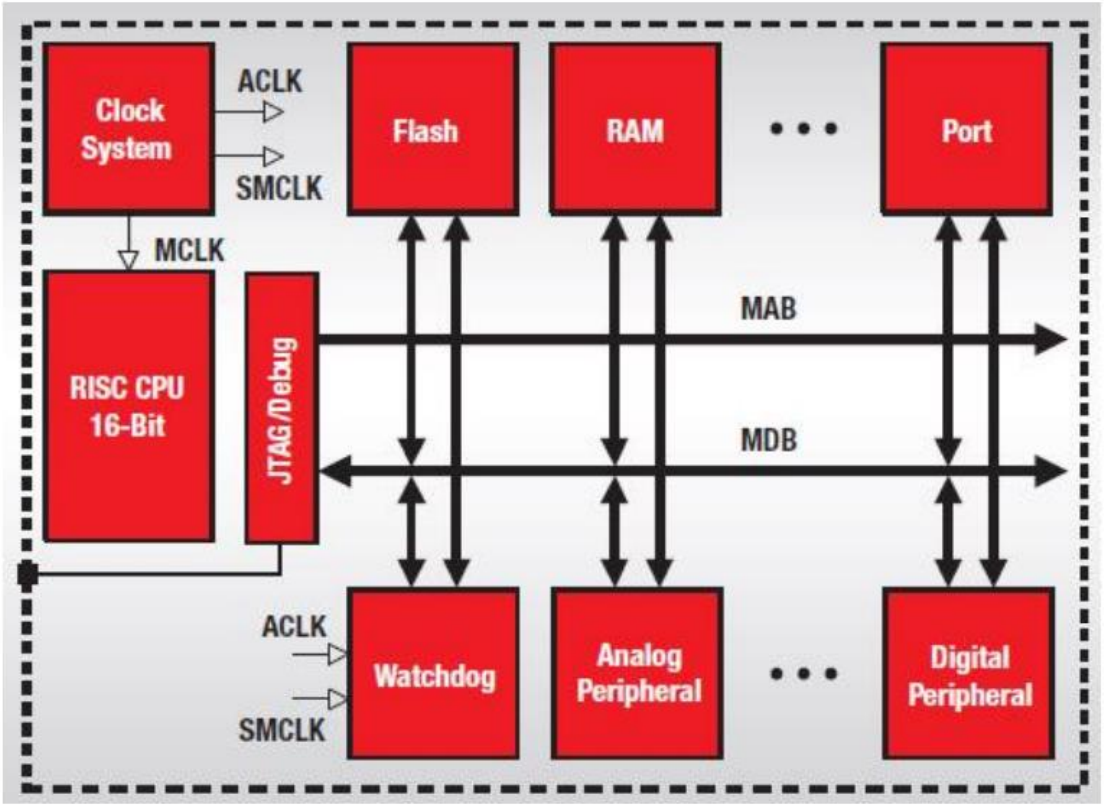
- 27 temel (core) instruction- hepsi unique opcode a sahipler,
- 16-bit veri yolu
- 16-bit registers
- 16-bit adres yolu
- 7 adresleme modu
- RISC CPU mimarisi olduğundan azaltılmış komut seti
- Hafıza organizasyonu açısından Von Neumann mimarisinde
- MAB-memory address bus ve MDB-memory data bus ile CPU diğer bellek ve birimlere erişir
- İçerisinde klasik haline gelmiş bir çok modülleri barındırmaktadır
  - I2C: 2 adet pin üzerinden iletişim kurmayı sağlayan bir yazılım protokolüdür. Sadece 2 pin ayrılarak aynı hat üzerine birçok RAM , EEPROM , RTC vb. parça bağlanıp kullanılabilir bu da fazladan pin ihtiyacını ortadan kaldırır.
  - SPI (Serial Peripheral Interface): Full-dublex olarak çalışır-veri alıp gönderme eş zamanlıdır.
  - USART(Universal Asynchronous Receiver Transmitter): Haberleşme yapmak için belirli bir clock'a ihtiyaç duyulmaz. Veri herhangi bir anda iletilebilir. Belirli standartlar kullanılarak gerçekleştirilir ve Senkron haberleşmeye göre daha yavaş bir iletim olur
  - ADC gibi

MSP430

	MSP430G2201	MSP430G22011	MSP430G2221	MSP430G2231
Program Memory(kB)	***2***	***2***	***2***	***2***
SRAM(Bytes)	***128***	***128***	***128***	***128***
I/O Pins	***10***	***10***	***10***	***10***
16-bit Timers	***1***	***1***	***1***	***1***
Capture/Compare Registers	***2***	***2***	***2***	***2***
USI:I2C/SPI				***1***
ADC Channels				***8***
ADC Resolutions(Bits)				***10***

MSP430G2x21, MSP430G2x31 Ailesinin Genel Özellikleri

MSP430



MSP430 Temel Blok Diyagramı

- MSP430 birçok osilatör ve clock seçeneğine sahiptir. Kendi içerisinde CPU ve diğer çevresel birimlerine ACLK, MCLK ve SMCLK clock sinyallerini sağlar
- ACLK (Auxiliary CLocK) : Düşük frekans ile çalışmayı talep eden çevresel birimleri beslemek için kullanılır.
- MCLK (Master CLocK): CPU ve sistemin genel clock ihtiyacını karşılar.
- SMCLK (Sub-system Master CLocK): Çevresel birimlerin (örneğin SPI, UART vb.) clock ihtiyacını karşılar.
- Bu clock sinyalleri, düşük güç modları (LPM-Low Power Mode) ile kapatılarak sistemin çektiği akım dolayısıyla güç azaltılabilir.

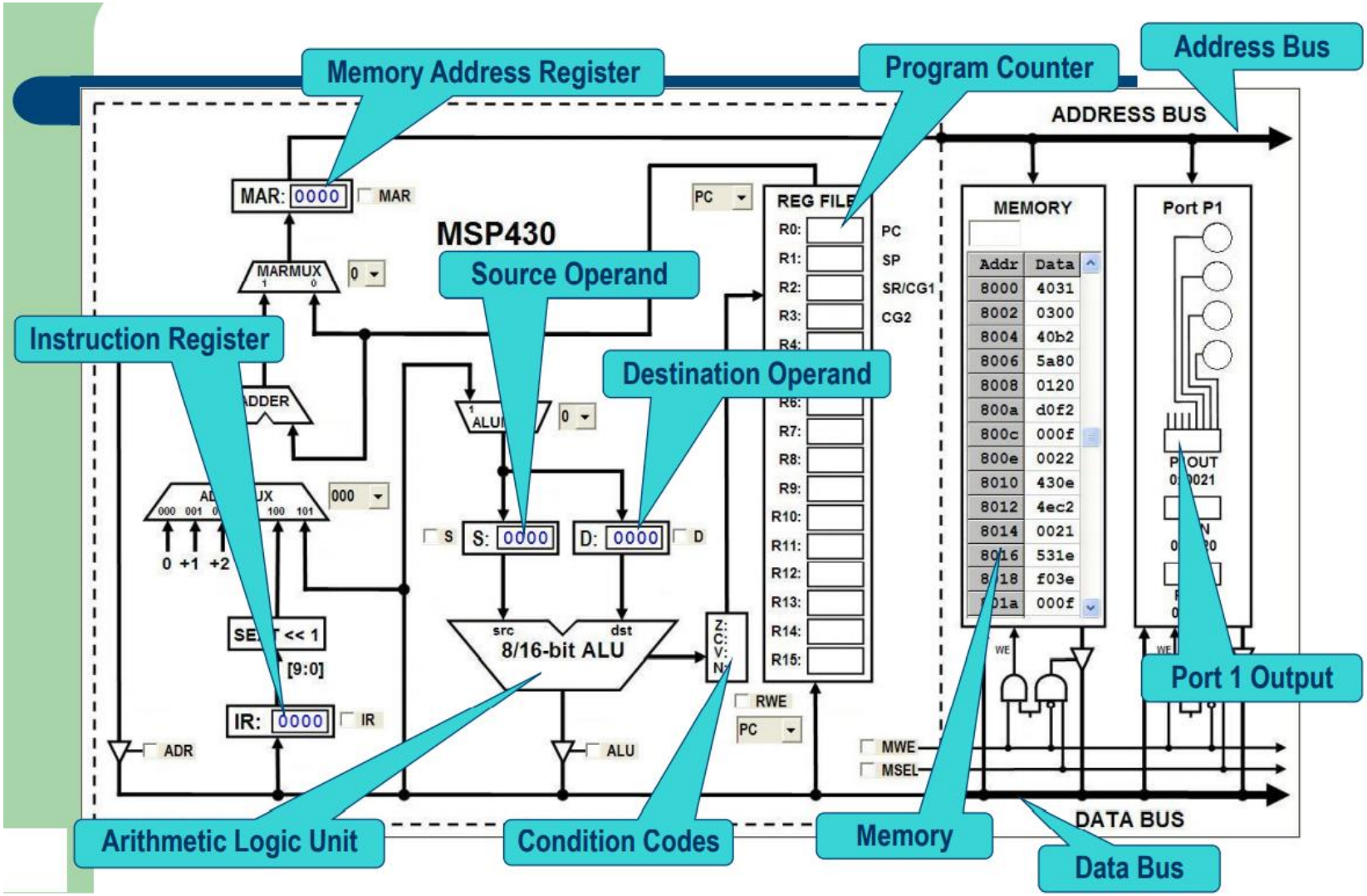
MSP430

- ACLK (Auxiliary CLocK) : 10 KHz VLO veya 32.768 KHz düşük frekanslı harici bir kristal (LFXT) osilatöründen sentezlenebilir.
- MCLK (Master CLocK): VLO/LFXT, DCO veya destekliyor ise yüksek frekanslı kristal XT (ya da HFXT) ile sentezlenebilir.
- SMCLK (Sub-system Master CLocK):DCO veya XT ile sentezlenebilir.

Sistem clock sinyalerini sentezlemek için MSP430 aşağıdaki osilatör seçeneklerini sunar

- VLO (Very Low-frequency Oscillator): MCU bünyesinde bulunan tipik 10 KHz frekanslı düşük güç tüketen internal (dahili) osilatördür.
- LFXT (Low-Frequency XTAL): Adından da anlaşılacağı üzere düşük frekanslı harici bir kristal ve (eğer gerekli ise) yük kapasitörleri ile ilgili pinlere bağlanması ile gerçekleştirilir. VLO ile karşılaştırıldığında oldukça stabildir ve zamanlama hassasiyetinin gerektiği devrelerde kullanılır. Düşük güç tüketir. Genel olarak 32.768 KHz kristal kullanılır.
- DCO (Digitally Controlled Oscillator): Yazılımsal olarak frekansı ayarlanabilen dahili osilatördür. MCU'in yüksek frekanslar ile çalışmasını istiyorsak MCLK ve SMCLK clock sinyallerini DCO'den sentezleyebiliriz. Frekansı 16 Mhz'e kadar çıkabilmektedir.
- XT (ya da HFXT (High-Frequency XTAL)): İlgili pinlere yüksek frekanslı kristal ve yük kapasitörlerinin bağlanmasıyla oluşturulur.

MSP430


Memory Address Register
Program Counter
Address Bus
Source Operand
Instruction Register
Destination Operand
Port 1 Output
Arithmetic Logic Unit
Condition Codes
Memory
Data Bus
ADDRESS BUS
DATA BUS
MSP430
MAR: 0000
MAR
MARMUX
PC
REG FILE
R0:
R1:
R2:
R3:
R4:
R5:
R6:
R7:
R8:
R9:
R10:
R11:
R12:
R13:
R14:
R15:
PC
SP
SR/CG1
CG2
MEMORY
Addr
Data
8000 4031
8002 0300
8004 40b2
8006 5a80
8008 0120
800a d0f2
800c 000f
800e 0022
8010 430e
8012 4ec2
8014 0021
8016 531e
8018 f03e
801a 000f
Port P1
S: 0000
D: 0000
S
D
0 +1 +2
000
[9:0]
IR: 0000
IR
ADR
ALU
src
dst
8/16-bit ALU
Z:
C:
V:
N:
RWE
MWE
MSEL
WE

MSP430

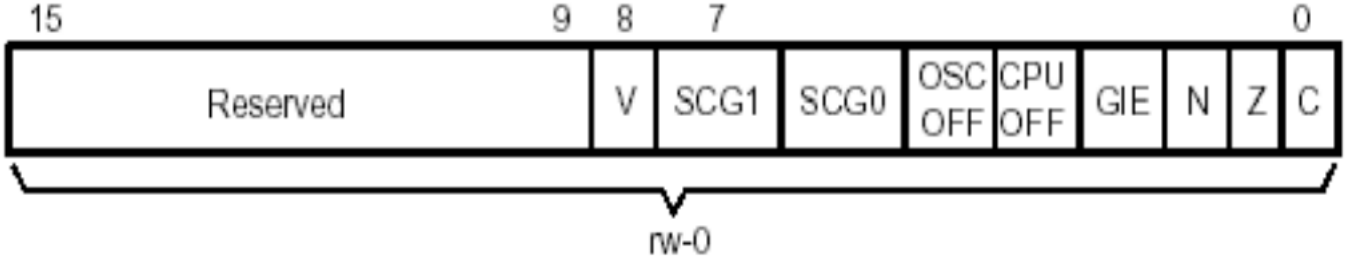
- 27 temel (core) instruction
- 16-bit veri yolu
- 16 tane 16-bit registers
- 16-bit adres yolu
- 7 adresleme modu
- 2^16 =64KB bellek adresleme alanı
- MSP430 CPU su modern programlama tekniklerinin kullanımına izin verir:
  - jump adreslerinin hesaplanması,
  - tablolarda veri işleme,
  - yüksek seviyeli dillerin kullanımı (C gibi)
- Her instruction her adresleme modlarıyla kullanılır,
- Word ve byte adresleme

MSP430

- 16 tane 16-bit registers
- **R0- (PC - Program Counter):** fetch edilecek bir sonraki instruction ı daima gösterir
  - Her komut çift sayıda byte kaplar.
  - Fetch işlemiden sonra, PC register 2, 4, veya 6 arttırılır.
- **R1- (SP-Stack Pointer):** Kesme-yığındaki dönüş adreslerini saklar
  - Her zaman çift adresi işaret eder,
  - RAM aracılığıyla yığın boyutu büyür bu nedenle SP geçerli bir RAM adresiyle başlatılmalıdır

MSP430

- **R2- (SR/CG1(constant-generator 1)-Status Register):**
  - Saat seçimleri, kesme enable/disable edilmesi,



V	Overflow bit – set when arithmetic operation overflows the signed-variable range.
SCG1	System clock generator 1 – turns off the SMCLK.
SCG0	System clock generator 0 – turns off the DCO dc generator.
OSCOFF	Oscillator off – turns off the LFXT1 crystal oscillator.
CPUOFF	CPU off – turns off the CPU.
GIE	General interrupt enable – enables maskable interrupts.
N	Negative bit – set when the result of a byte or word operation is negative.
Z	Zero bit – set when the result of a byte or word operation is 0.
C	Carry bit – set when the result of a byte or word operation produces a carry.

MSP430

- **R3- (CG2(constant-generator 2)**
  - R2 ve R3 kullanılarak 6 farklı sabit üretilebilir

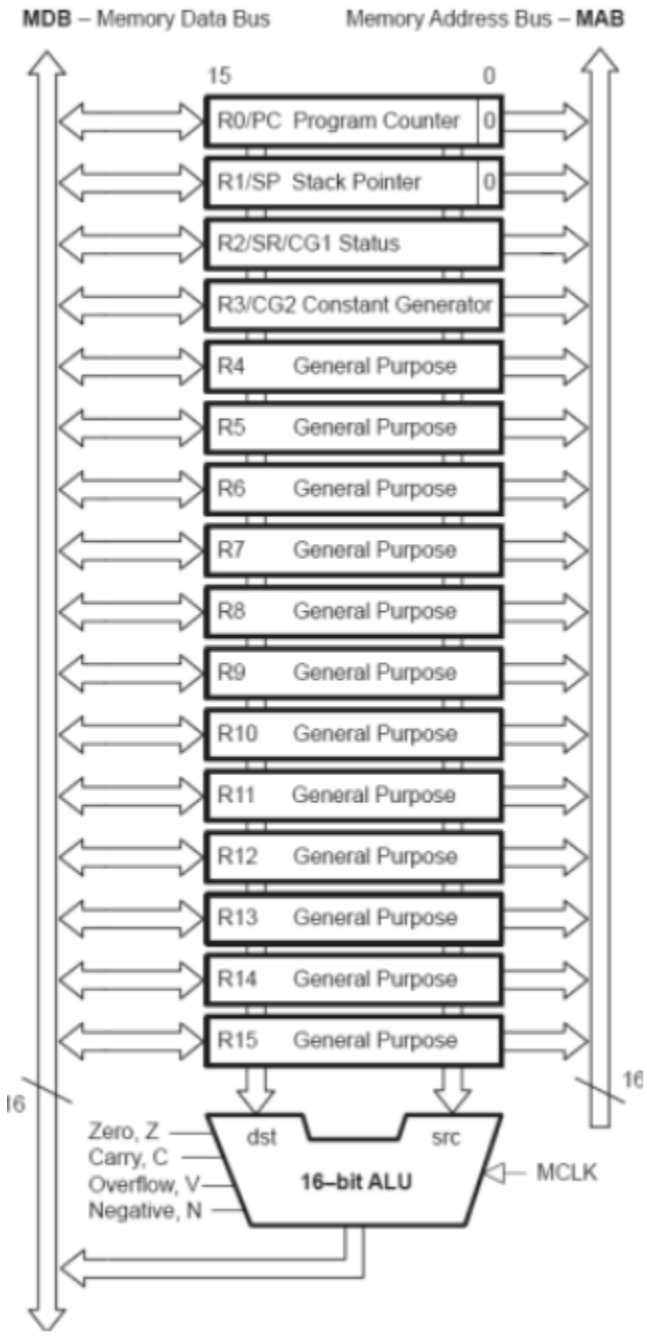
Register	As	Constant	Remarks
R2	00	-	Register mode
R2	01	(0)	Absolute mode
R2	10	00004h	+4, bit processing
R2	11	00008h	+8, bit processing
R3	00	00000h	0, word processing
R3	01	00001h	+1
R3	10	00002h	+2, bit processing
R3	11	0FFFFh	-1, word processing

MSP430

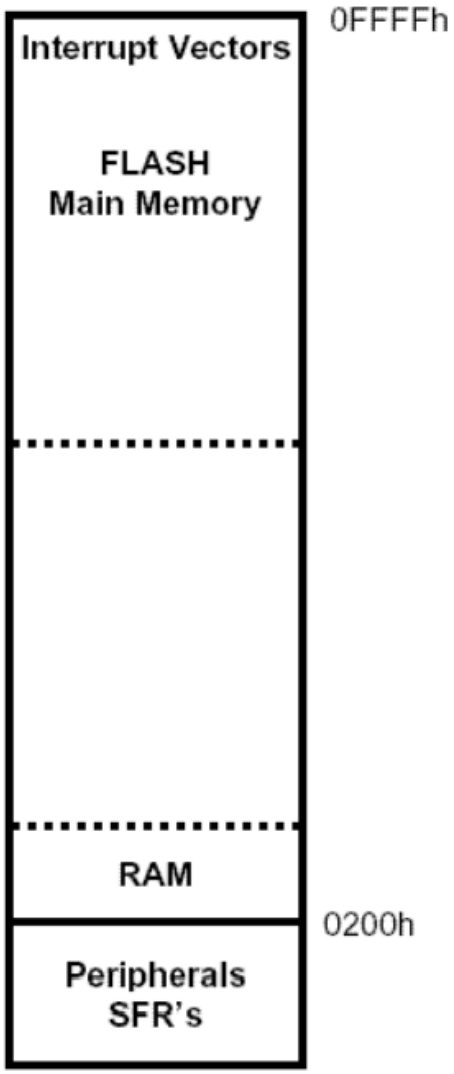
- **R4-R15 – Genel Amaçlı Registerlar**
  - Instruction formatları byte veya word olarak erişilebilirler
  - Bu registerlara da byte veya word olarak erişilebilir

MSP430

- 16-bit ALU
  - Aritmetik ve lojik işlemleri yapar
- Instruction execution durumu aşağıdaki bayrakların durumunu etkiler
  - Zero (Z)
  - Carry (C)
  - Overflow(V)
  - Negative (N)
- MCLK sinyali CPUnun genel clock ihtiyacını karşılar



MSP430



- 64 KB bellek haritası
- Data ve çevre birimler için aynı instructionlar
- Program ve veri Flash veya RAM içindedir
- Modern programlama teknikleri için tasarlanmıştır; pointer ve hızlı look-up tablellar gibi

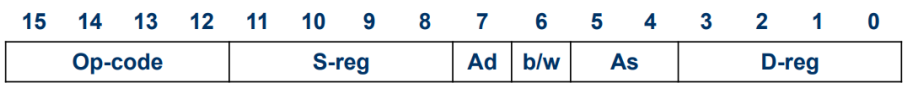
MSP430

- Opcode: instruction ne iş yapar
- Source Operand (SO): kaynak operand
- Destination Operand (DO): hedef operand, sonuçlar burada depolanır
- Instructionları encode etmek için 3 format vardır
  - Double operand (.W): word data erişimi sağlar, defaultta instructionlar word data olarak işlenirler
  - Single operand (.B): byte data erişimi sağlar
  - jumps

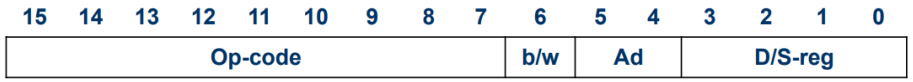
MSP430

- src: As ve S-reg'de tanımlandığı gibi kaynak operand adresi
- dst: Ad ve D-reg'de tanımlandığı gibi hedef operand adresi
- As: kaynak operand tarafından kullanılan adresleme modunu tanımlamak için kullanılan adresleme bitleri
- S-reg: kaynak operandi tarafından kullanılan register
- Ad: hedef operande tarafından kullanılan adresleme modunu tanımlamak için kullanılan adresleme bitleri
- D-reg: hedef operand tarafından kullanılan register
- b / w: kelime veya byte erişim tanım biti.

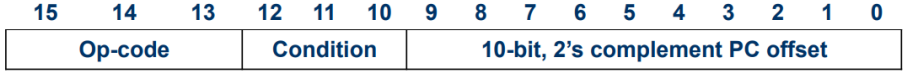
- Format I: Instructions with two operands:



- Format II: Instruction with one operand:



- Format II: Jump instructions:




```
; Format I Source and Destination
```

Op-Code	Source-Register	Ad	B/W	As	Destination-Register

```
5405        add.w   R4,R5                ; R4+R5=R5 xxxx
5445        add.b   R4,R5                ; R4+R5=R5 00xx
```

```
; Format II Destination Only
```

Op-Code	B/W	Ad	D/S- Register

```
6404        rlc.w   R4                   ;
6444        rlc.b   R4                   ;
```

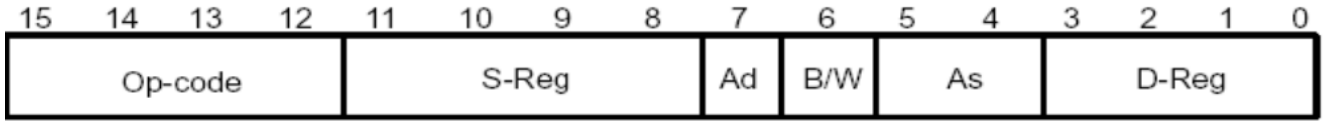
```
; Format III There are 8(Un)conditional Jumps
```

Op-Code	Condition	10-bit PC offset

```
3c28        jmp     Loop_1               ; Goto Loop_1
```

MSP430

- **Double operand instructions**



Mnemonic	S-Reg, D-Reg	Operation	Status Bits			
			V	N	Z	C
MOV(.B)	src,dst	src → dst	–	–	–	–
ADD(.B)	src,dst	src + dst → dst	*	*	*	*
ADDC(.B)	src,dst	src + dst + C → dst	*	*	*	*
SUB(.B)	src,dst	dst + .not.src + 1 → dst	*	*	*	*
SUBC(.B)	src,dst	dst + .not.src + C → dst	*	*	*	*
CMP(.B)	src,dst	dst – src	*	*	*	*
DADD(.B)	src,dst	src + dst + C → dst (decimally)	*	*	*	*
BIT(.B)	src,dst	src .and. dst	0	*	*	*
BIC(.B)	src,dst	.not.src .and. dst → dst	–	–	–	–
BIS(.B)	src,dst	src .or. dst → dst	–	–	–	–
XOR(.B)	src,dst	src .xor. dst → dst	*	*	*	*
AND(.B)	src,dst	src .and. dst → dst	0	*	*	*

source:kaynak
destination: hedef

MSP430

- **Double operand instructions**

Mnemonic	Operation	Description
Arithmetic instructions		
ADD(.B or .W) src,dst	src+dst→dst	Add source to destination
ADDC(.B or .W) src,dst	src+dst+C→dst	Add source and carry to destination
DADD(.B or .W) src,dst	src+dst+C→dst (dec)	Decimal add source and carry to destination
SUB(.B or .W) src,dst	dst+.not.src+1→dst	Subtract source from destination
SUBC(.B or .W) src,dst	dst+.not.src+C→dst	Subtract source and not carry from destination
Logical and register control instructions		
AND(.B or .W) src,dst	src.and.dst→dst	AND source with destination
BIC(.B or .W) src,dst	.not.src.and.dst→dst	Clear bits in destination
BIS(.B or .W) src,dst	src.or.dst→dst	Set bits in destination
BIT(.B or .W) src,dst	src.and.dst	Test bits in destination
XOR(.B or .W) src,dst	src.xor.dst→dst	XOR source with destination
Data instructions		
CMP(.B or .W) src,dst	dst-src	Compare source to destination
MOV(.B or .W) src,dst	src→dst	Move source to destination

MSP430

Örnek: Double operand instructions

- Registerın içeriğini registera kopyala
  - Assembly: **mov.w r5,r4**
  - Instruction code: **0x4504**

Op-code mov	S-reg r5	Ad Register	b/w 16-bits	As Register	D-reg r4
0 1 0 0	0 1 0 1	0	0	0 0	0 1 0 0

- r5 registerının içindeki 16-bit 2'nin complementi word r4 registerına kopyalar

MSP430

Örnek: Double operand instructions

- Registerın içeriğini PC-relative memory adres gözüne kopyala
  - Assembly: **mov.w r5,TONI**
  - Instruction code: **0x4580**

Op-code mov	S-reg r5	Ad Symbolic	b/w 16-bits	As Register	D-reg PC
0 1 0 0	0 1 0 1	1	0	0 0	0 0 0 0
2's complement PC-relative destination index					

- Komut bir belleği talimat verir; r5 registerının içindeki 16-bit 2'nin complementi word, (PC+destination index) bellek adresine kopyalanır

MSP430

Örnek: Double operand instructions

- PC-relative memory içeriğini başka bir PC-relative memory adres gözüne kopyala

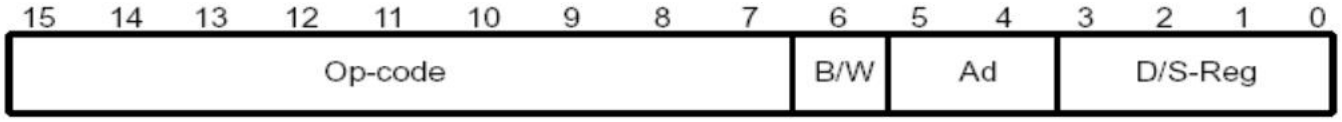
  - Assembly: **mov.b EDEN,TONI**
  - Instruction code: **0x40d0**

Op-code mov	S-reg PC	Ad Symbolic	b/w 8-bits	As Symbolic	D-reg PC
0 1 0 0	0 0 0 0	1	1	0 1	0 0 0 0
2's complement PC-relative source index					
2's complement PC-relative destination index					

- EDEN (source index+PC) bellek gözündeki 8-bit içeriği, TONI (destination index+PC) bellek gözüne kopyalar

MSP430

- **Single operand instructions**



Mnemonic	S-Reg, D-Reg	Operation	Status Bits			
			V	N	Z	C
RRC(.B)	dst	C → MSB →.....LSB → C	*	*	*	*
RRA(.B)	dst	MSB → MSB →....LSB → C	0	*	*	*
PUSH(.B)	src	SP – 2 → SP, src → @SP	–	–	–	–
SWPB	dst	Swap bytes	–	–	–	–
CALL	dst	SP – 2 → SP, PC+2 → @SP	–	–	–	–
		dst → PC				
RETI		TOS → SR, SP + 2 → SP	*	*	*	*
		TOS → PC,SP + 2 → SP				
SXT	dst	Bit 7 → Bit 8.....Bit 15	0	*	*	*

MSP430

- **Single operand instructions**

Mnemonic	Operation	Description
Logical and register control instructions		
`RRA(.B or .W) dst`	MSB→MSB→… LSB→C	Roll destination right
`RRC(.B or .W) dst`	C→MSB→…LSB→C	Roll destination right through carry
`SWPB( or .W) dst`	Swap bytes	Swap bytes in destination
`SXT dst`	bit 7→bit 8…bit 15	Sign extend destination
`PUSH(.B or .W) src`	SP-2→SP, src→@SP	Push source on stack
Program flow control instructions		
`CALL(.B or .W) dst`	SP-2→SP, PC+2→@SP dst→PC	Subroutine call to destination
`RETI`	@SP+→SR, @SP+→SP	Return from interrupt

MSP430

Örnek: Single operand instructions

- r5 registerının içeriğini mantıksal olarak sağa kaydır (durum elde (C) registerı aracılığıyla)

  - Assembly: **rrc.w r5**
  - Instruction code: **0x1005**

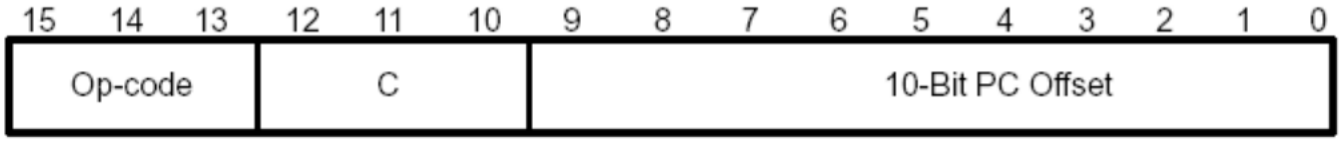
Op-code rrc	*b/w 16-bits*	*Ad Register*	*D-reg r5*
0 0 0 1 0 0 0 0 0	0	0 0	0 1 0 1

- 16-bit r5 registerının içindeki wordu 1 bit sağa kaydır (2'ye bölme) MSB içeriği C, C nin içeriği LSB olur

Zero (Z)
Carry (C)
Overflow(V)
Negative (N)

MSP430

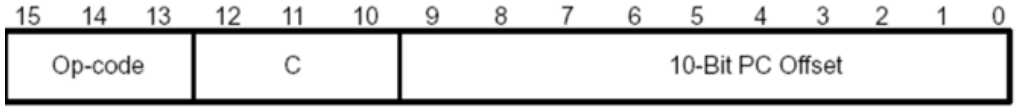
Jump instructions



Mnemonic	S-Reg, D-Reg	Operation	
JEQ/JZ	Label	Jump to label if zero bit is set	Jump if equal
JNE/JNZ	Label	Jump to label if zero bit is reset	Jump if not equal
JC	Label	Jump to label if carry bit is set	Jump if carry=1
JNC	Label	Jump to label if carry bit is reset	Jump if carry=0
JN	Label	Jump to label if negative bit is set	Jump if nagative=1
JGE	Label	Jump to label if (N .XOR. V) = 0	Jump if greater than or equal
JL	Label	Jump to label if (N .XOR. V) = 1	Jump if lower
JMP	Label	Jump to label unconditionally	Unconditional jump (koşulsuz)

MSP430

Jump Instructions



- Dallanma komutları programı, programın başka bir parçasına yönlendirmek için kullanılır.

C:Condition 3-bittir

- 000: jump if not equal
- 001: jump if equal
- 010: jump if carry flag equal to zero
- 011: jump if carry flag equal to one
- 100: jump if negative (N = 1)
- 101: jump if greater than or equal (N = V)
- 110: jump if lower (N ≠ V)
- 111: unconditional jump

MSP430

Emulated Instructions

- 27 ana komutun yanı sıra 24 emulated komut içerir,
- Bu komutlar kodu okuyup yazmayı kolaylaştırır, ancak kendi opecode ları yoktur
- CPU tarafından otomatik olarak core instructionlarla değiştirir

MSP430

Emulated Instructions

Mnemonic	Operation	Emulation	Description
Arithmetic instructions			
ADC(.B or .W) dst	dst+C→dst	ADDC(.B or .W) #0,dst	Add carry to destination
DADC(.B or .W) dst	dst + C → dst (decimally)	DADD(.B or .W) #0,dst	Decimal add carry to destination
DEC(.B or .W) dst	dst-1→dst	SUB(.B or .W) #1,dst	Decrement destination
DECD(.B or .W) dst	dst-2→dst	SUB(.B or .W) #2,dst	Decrement destination twice
INC(.B or .W) dst	dst+1→dst	ADD(.B or .W) #1,dst	Increment destination
INCD(.B or .W) dst	dst+2→dst	ADD(.B or .W) #2,dst	Increment destination twice
SBC(.B or .W) dst	dst+0FFFFh+C→dst dst+0FFh→dst	SUBC(.B or .W) #0,dst	Subtract source and borrow /.NOT. carry from dest.

MSP430

Emulated Instructions

Mnemonic	Operation	Emulation	Description
Logical and register control instructions			
INV(.B or .W) dst	.NOT.dst→dst	XOR(.B or .W) #0(FF)FFh,dst	Invert bits in destination
RLA(.B or .W) dst	C←MSB←MSB-1 LSB+1←LSB←0	ADD(.B or .W) dst,dst	Rotate left arithmetically
RLC(.B or .W) dst	C←MSB←MSB-1 LSB+1←LSB←C	ADDC(.B or .W) dst,dst	Rotate left through carry
Program flow control			
BR dst	dst→PC	MOV dst,PC	Branch to destination
DINT	0→GIE	BIC #8,SR	Disable (general) interrupts
EINT	1→GIE	BIS #8,SR	Enable (general) interrupts
NOP	None	MOV #0,R3	No operation
RET	@SP→PC SP+2→SP	MOV @SP+,PC	Return from subroutine

MSP430

Emulated Instructions

Mnemonic	Operation	Emulation	Description
Data instructions			
CLR(.B or .W) dst	0→dst	MOV(.B or .W) #0,dst	Clear destination
CLRC	0→C	BIC #1,SR	Clear carry flag
CLRN	0→N	BIC #4,SR	Clear negative flag
CLRZ	0→Z	BIC #2,SR	Clear zero flag
POP(.B or .W) dst	@SP→temp SP+2→SP temp→dst	MOV(.B or .W) @SP +,dst	Pop byte/word from stack to destination
SETC	1→C	BIS #1,SR	Set carry flag
SETN	1→N	BIS #4,SR	Set negative flag
SETZ	1→Z	BIS #2,SR	Set zero flag
TST(.B or .W) dst	dst + 0FFFFh + 1 dst + 0FFh + 1	CMP(.B or .W) #0,dst	Test destination

MSP430

Emulated Instructions

Örnek: Emulated Instructions

- `CLR R5`

- Instruction code: 0x4305

Op-code mov	S-reg r3	Ad Register	b/w 16-bits	As Register	D-reg r5
0 1 0 0	0 0 1 1	0	0	0 0	0 1 0 1

Bu instruction R3 = #0 atanarak, MOV R3,R5, komutuna eşittir.

MSP430

- 27 core instruction

Table 1a: The complete MSP430 instruction set of 27 core instructions

core instruction mnemonics	core instruction binary															
Single-operand arithmetic	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	1	0	0	opcode			B/W	As		source			
RRC Rotate right through carry	0	0	0	1	0	0	0	0	0	B/W	As		source			
SWPB Swap bytes	0	0	0	1	0	0	0	0	1	0	As		source			
RRA Rotate right arithmetic	0	0	0	1	0	0	0	1	0	B/W	As		source			
SXT Sign extend byte to word	0	0	0	1	0	0	0	1	1	0	As		source			
PUSH Push value onto stack	0	0	0	1	0	0	1	0	0	B/W	As		source			
CALL Subroutine call; push PC and move source to PC	0	0	0	1	0	0	1	0	1	0	As		source			
RETI Return from interrupt; pop SR then pop PC	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0
Conditional jump; PC = PC + 2×offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	1	condition			10-bit signed offset									
JNE/JNZ Jump if not equal/zero	0	0	1	0	0	0	10-bit signed offset									
JEQ/JZ Jump if equal/zero	0	0	1	0	0	1	10-bit signed offset									
JNC/JLO Jump if no carry/lower	0	0	1	0	1	0	10-bit signed offset									
JC/JHS Jump if carry/higher or same	0	0	1	0	1	1	10-bit signed offset									
JN Jump if negative	0	0	1	1	0	0	10-bit signed offset									
JGE Jump if greater or equal (N == V)	0	0	1	1	0	1	10-bit signed offset									
JL Jump if less (N != V)	0	0	1	1	1	0	10-bit signed offset									
JMP Jump (unconditionally)	0	0	1	1	1	1	10-bit signed offset									
Two-operand arithmetic	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	opcode				source				Ad	B/W	As		destination			
MOV Move source to destination	0	1	0	0	source				Ad	B/W	As		destination			
ADD Add source to destination	0	1	0	1	source				Ad	B/W	As		destination			
ADDC Add w/carry: dst += (src+C)	0	1	1	0	source				Ad	B/W	As		destination			
SUBC Subtract w/ carry: dst -= (src+C)	0	1	1	1	source				Ad	B/W	As		destination			
SUB Subtract; dst -= src	1	0	0	0	source				Ad	B/W	As		destination			
CMP Compare; (dst-src); discard result	1	0	0	1	source				Ad	B/W	As		destination			
DADD Decimal (BCD) addition: dst += src	1	0	1	0	source				Ad	B/W	As		destination			
BIT Test bits; (dst & src); discard result	1	0	1	1	source				Ad	B/W	As		destination			
BIC Bit clear; dest &= ~src	1	1	0	0	source				Ad	B/W	As		destination			
BIS "Bit set" - logical OR; dst |= src	1	1	0	1	source				Ad	B/W	As		destination			
XOR Bitwise XOR; dst ^= src	1	1	1	0	source				Ad	B/W	As		destination			
AND Bitwise AND; dst &= src	1	1	1	1	source				Ad	B/W	As		destination			

MSP430

- Kaynak (source) adresleme modları:
- Kaynak adres için 4 temel adresleme modu
  - Rs – Register
  - x(Rs) - Indexed Register
  - @Rs - Register Indirect
  - @Rs+ - Indirect Auto-increment
- R0-R3 registerlarıyla, 3 ek adresleme modu elde edilir:
  - label - PC Relative, x(PC)
  - &label – Absolute, x(SR)
  - #n – Immediate, @PC+

MSP430

- Hedef adres için 2 temel adresleme modu
  - Rd - Register
  - x(Rd) Indexed Register
- R0-R2 registerlarıyla, 2 ek adresleme modu elde edilir:
  - label - PC Relative, x(PC)
  - &label – Absolute, x(SR)

MSP430
Adresleme Modları

- Register Mode (Rn)
  - mov.w r5,r6 ; move word from r5 to r6
- Indexed Mode x(Rn)
  - mov.b 3(r5),r6 ; move byte from ; M(3_{10}+r5) to r6
- Symbolic Mode (PC Relative)
  - mov.w Cnt,r6 ; move word ; M(Cnt+PC) to r6
- Absolute Mode (&label)
  - mov.w &Cnt,r6 ; move word ; M(Cnt) to r6

MSP430
Adresleme Modları

- Indirect Register Mode (@Rn)
  - mov.w @r5,r6 ; move word ; M(r5) to r6
- Indirect Autoincrement Mode (@Rn+)
  - mov.w @r5+,r6 ; move word ; M(r5) to r6 ; increment r5 by 2 (.w +2, .b +1)
- Immediate Mode (#n)
  - mov.w #100,r6 ; 100 -> r6

MSP430

- Adresleme Modları Özet

As/Ad	Addressing Mode	Syntax	Description
00/0	Register mode	Rn	Register contents are operand
01/1	Indexed mode	X(Rn)	(Rn + X) points to the operand. X is stored in the next word.
01/1	Symbolic mode	ADDR	(PC + X) points to the operand. X is stored in the next word. Indexed mode X(PC) is used.
01/1	Absolute mode	&ADDR	The word following the instruction contains the absolute address. X is stored in the next word. Indexed mode X(SR) is used.
10/–	Indirect register mode	@Rn	Rn is used as a pointer to the operand.
11/–	Indirect autoincrement	@Rn+	Rn is used as a pointer to the operand. Rn is incremented afterwards by 1 for .B instructions and by 2 for .W instructions.
11/–	Immediate mode	#N	The word following the instruction contains the immediate constant N. Indirect autoincrement mode @PC+ is used.

ADDRESS MODE	S	D	SYNTAX	EXAMPLE	OPERATION
Register	●	●	MOV Rs,Rd	MOV R10,R11	R10 --> R11
Indexed	●	●	MOV X(Rn),Y(Rm)	MOV 2(R5),6(R6)	M(2+R5)--> M(6+R6)
Symbolic (PC relative)	●	●	MOV EDE,TONI		M(EDE) --> M(TONI)
Absolute	●	●	MOV &MEM,&TCDAT		M(MEM) --> M(TCDAT)
Indirect	●		MOV @Rn,Y(Rm)	MOV @R10,Tab(R6)	M(R10) --> M(Tab+R6)
Indirect autoincrement	●		MOV @Rn+,Rm	MOV @R10+,R11	M(R10) --> R11 R10 + 2--> R10
Immediate	●		MOV #X,TONI	MOV #45,TONI	#45 --> M(TONI)

NOTE: S = source D = destination

CCS v10 Download

- https://software-dl.ti.com/ccs/esd/documents/ccs_downloads.html

Kaynaklar

- https://maker.robotistan.com/mikrodenetleyici-mikroislemci/
- https://www.ozgurakin.com.tr/download/5-Mikroislemciler_ve_Mikrodenetleyiciler.pdf
- https://slideplayer.biz.tr/slide/8852192/
- https://en.wikipedia.org/wiki/Transistor_count
- https://www.xtrlarge.com/2018/12/17/transistor-uretim-boyut-rekor/
- https://anakart.wordpress.com/seri-ve-paralel-portlar/
- https://www.youtube.com/watch?v=qTpLYq-TPnA
- https://www.slideserve.com/meghan-malone/the-msp430-instruction-set
- http://cnrgzgz.com/msp430-hakkinda/
- https://www.mcu-turkey.com/msp430-egitim-2/
- https://medium.com/@mesuttopuzlu/g%C3%B6m%C3%BCl%C3%BC-yaz%C4%B1l%C4%B1ma-msp430-ile-ba%C5%9Flamak-778ae1f24688