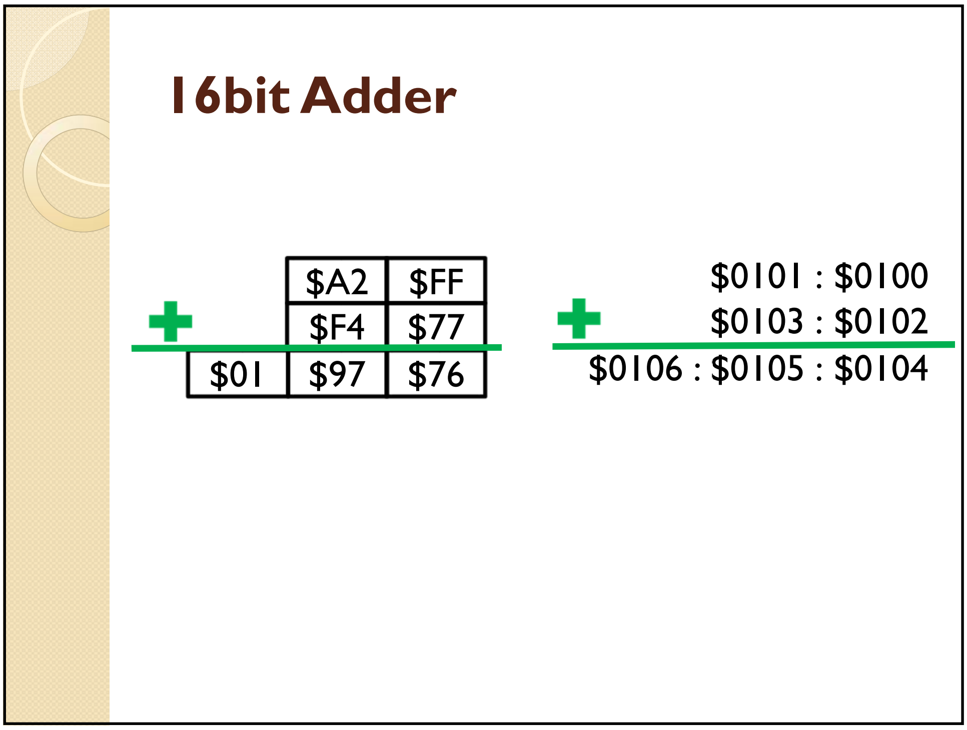
ECE375 Lab 5

TA: Youngbin Jin
School of Electrical Engineering and Computer Science
Oregon State University

Large Number Arithmetic

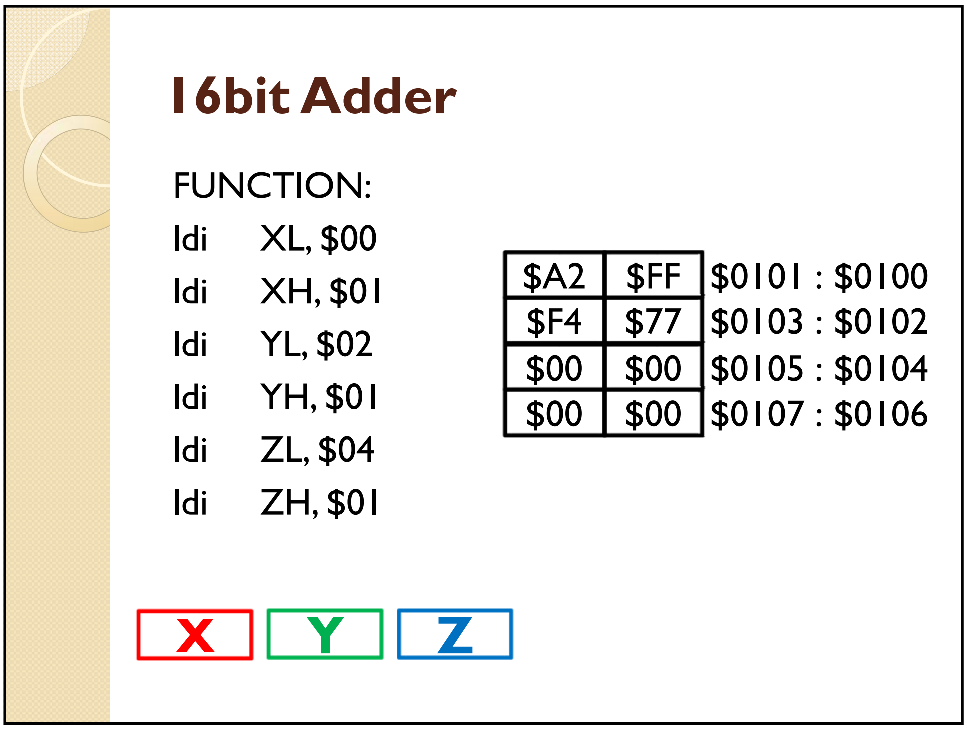
- AVR 8-bit instruction set
- Perform arithmetic operations on numbers that are larger than 8 bits
- Practice Memory Manipulation
- The Skeleton Code includes operands for ADD16, SUB16, MUL24, and Compound



16bit Adder

		$A2	$FF
+		$F4	$77
	$01	$97	$76

		$0101 :	$0100
+		$0103 :	$0102
	$0106 :	$0105 :	$0104

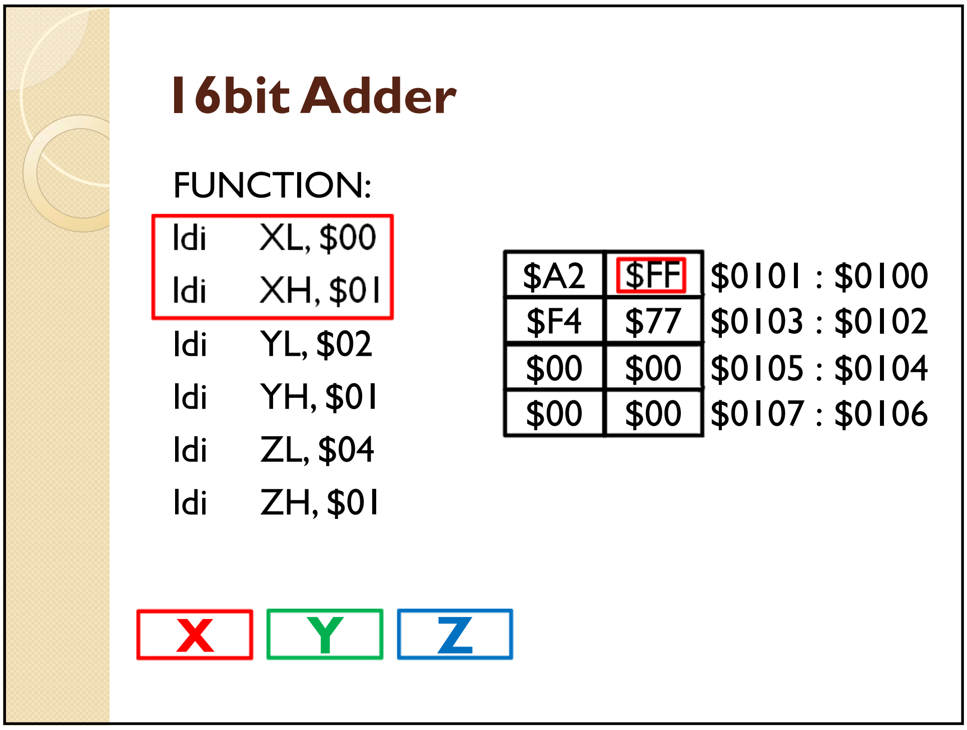


16bit Adder

```
FUNCTION:
ldi     XL, $00
ldi     XH, $01
ldi     YL, $02
ldi     YH, $01
ldi     ZL, $04
ldi     ZH, $01
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

X Y Z



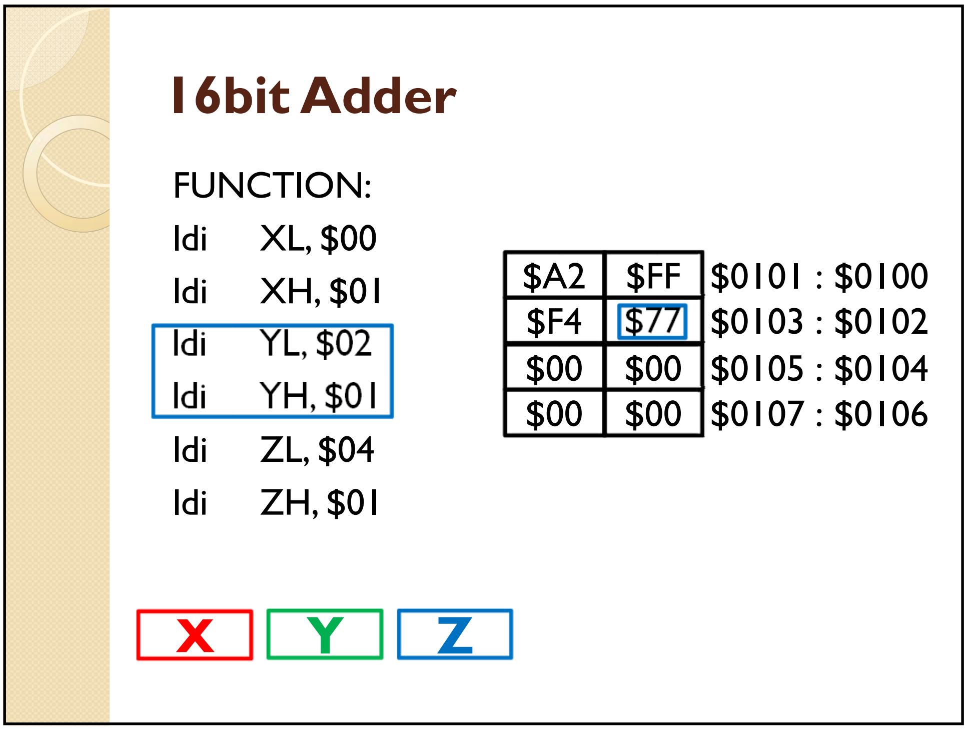
I 6bit Adder

FUNCTION:

```
ldi	XL, $00
ldi	XH, $0I
ldi	YL, $02
ldi	YH, $0I
ldi	ZL, $04
ldi	ZH, $0I
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

X Y Z



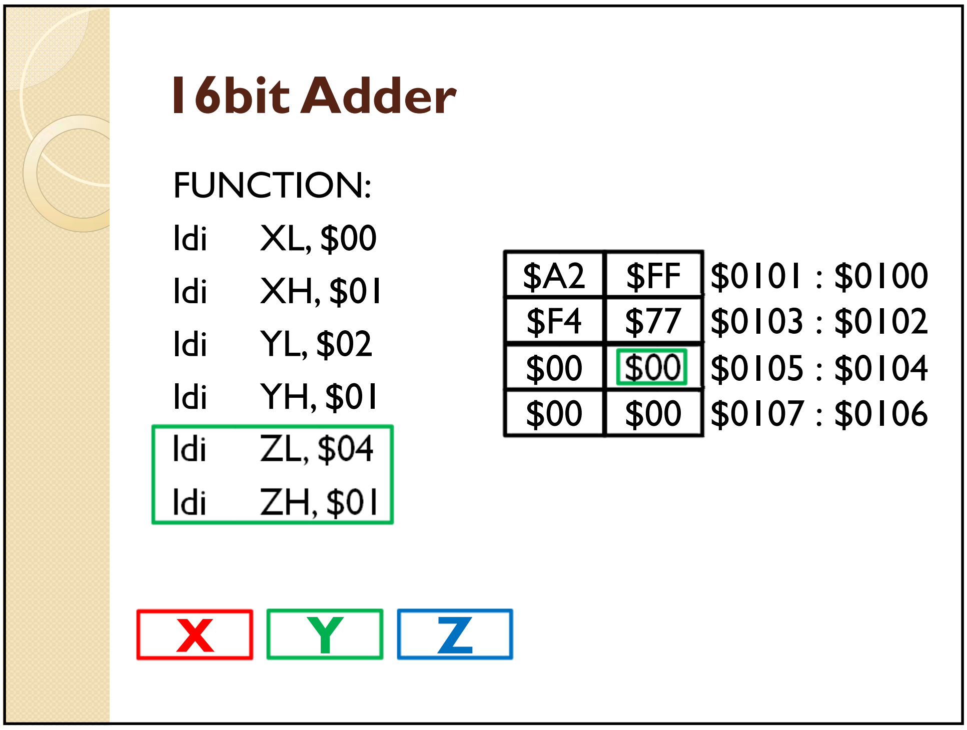
I 6bit Adder

FUNCTION:

```
ldi	XL, $00
ldi	XH, $0I
ldi	YL, $02
ldi	YH, $0I
ldi	ZL, $04
ldi	ZH, $0I
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

X Y Z



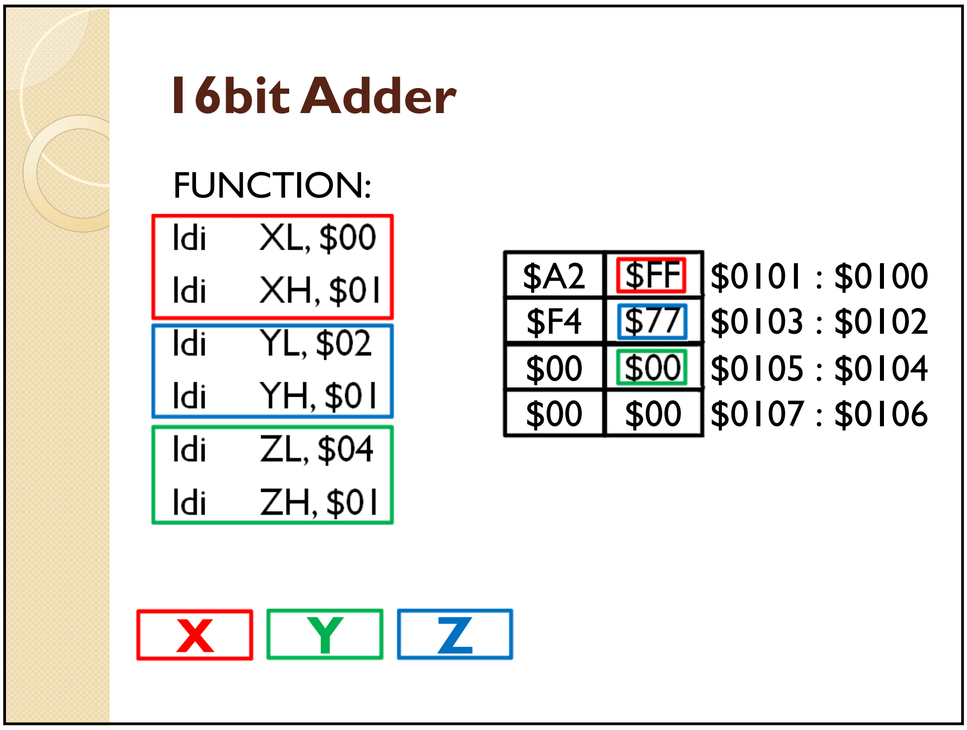
16bit Adder

FUNCTION:

```
ldi    XL, $00
ldi    XH, $01
ldi    YL, $02
ldi    YH, $01
ldi    ZL, $04
ldi    ZH, $01
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

X Y Z



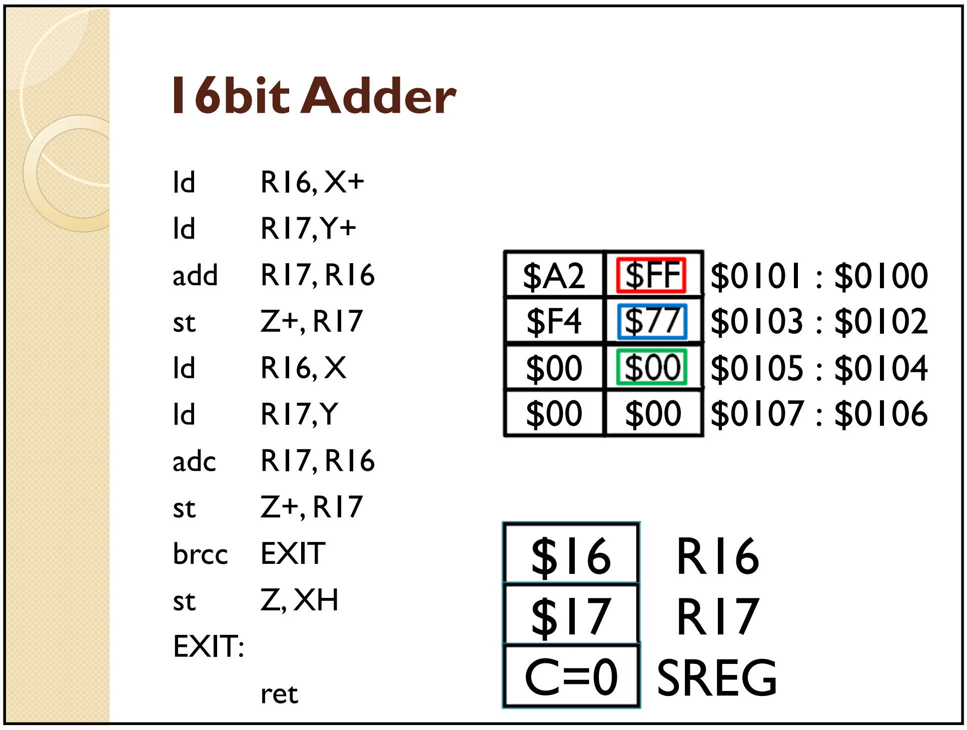
16bit Adder

FUNCTION:

```
ldi    XL, $00
ldi    XH, $01
ldi    YL, $02
ldi    YH, $01
ldi    ZL, $04
ldi    ZH, $01
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

X Y Z

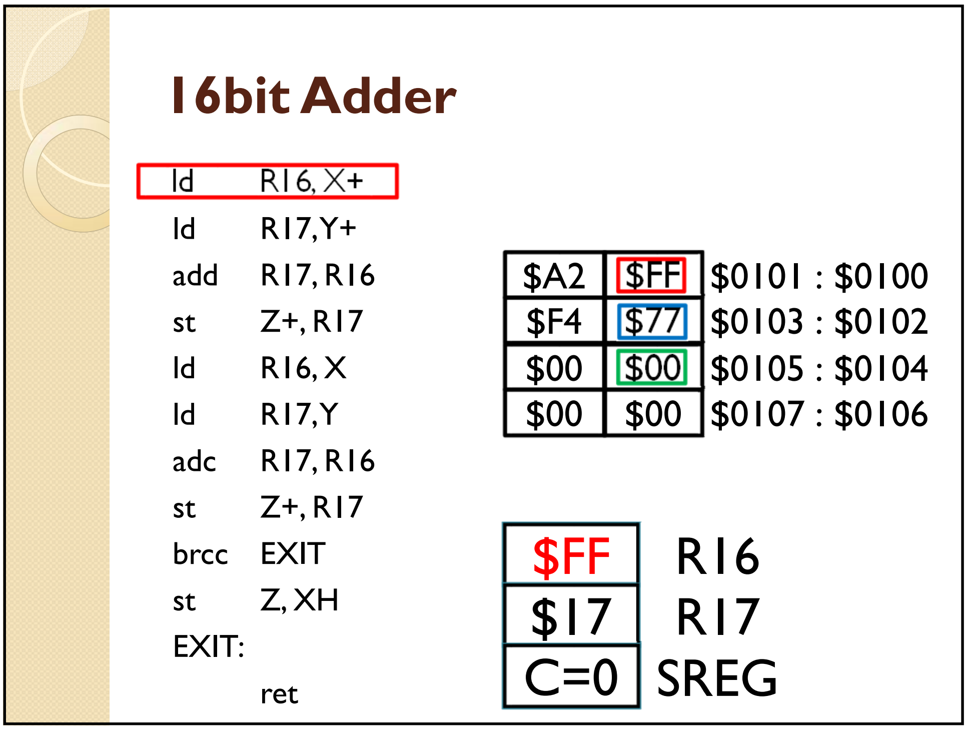


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$16	R16
$17	R17
C=0	SREG

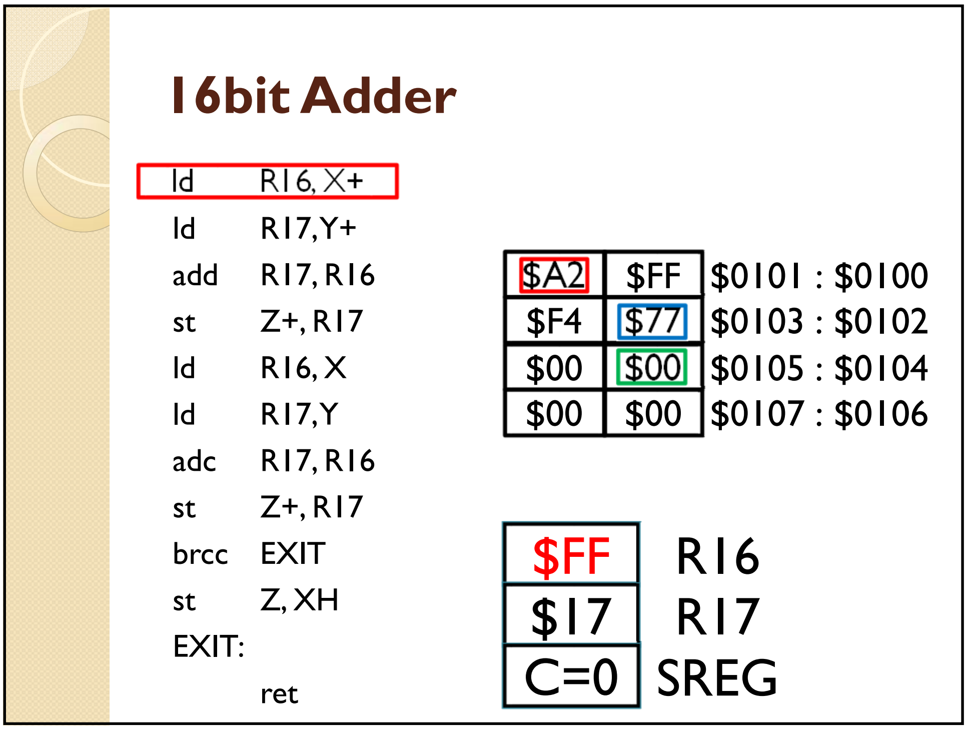


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$17	R17
C=0	SREG

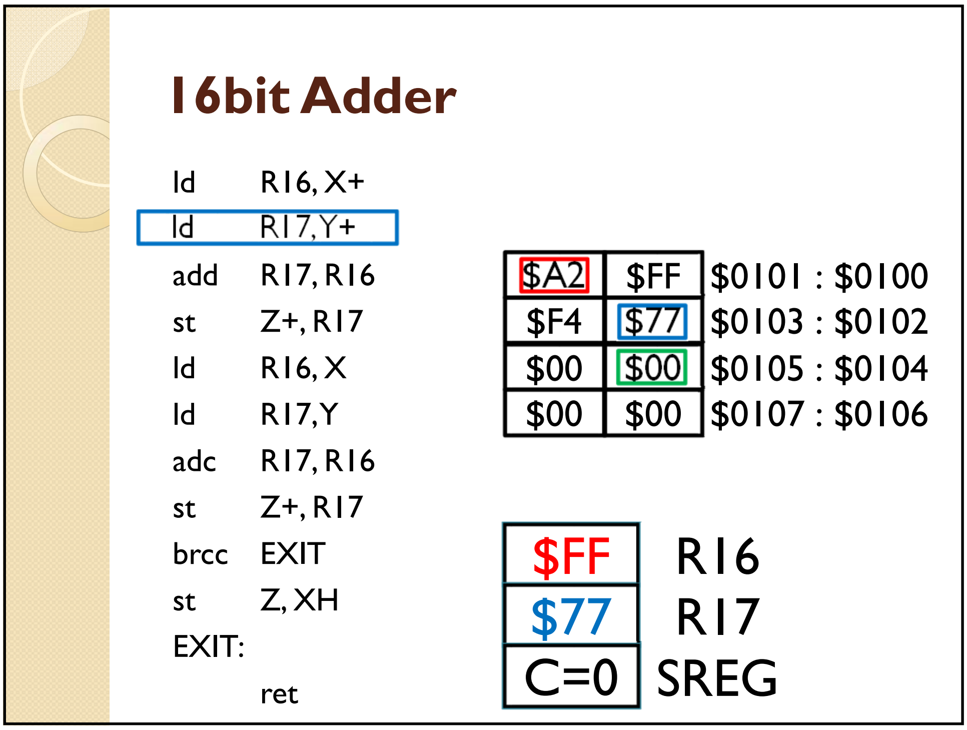


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$17	R17
C=0	SREG

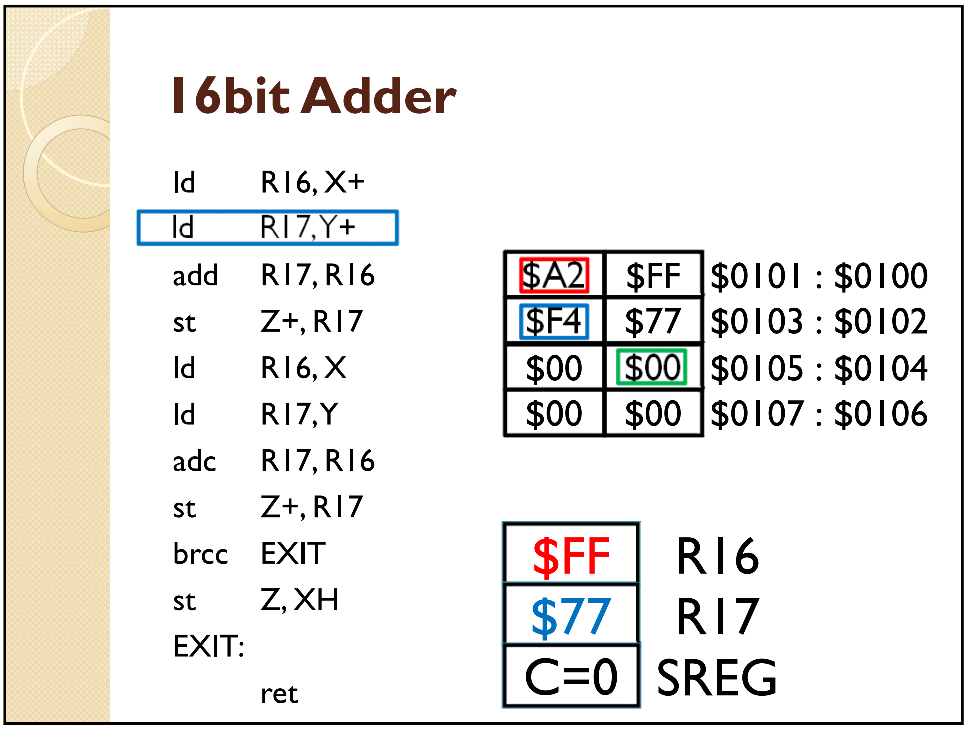


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$77	R17
C=0	SREG

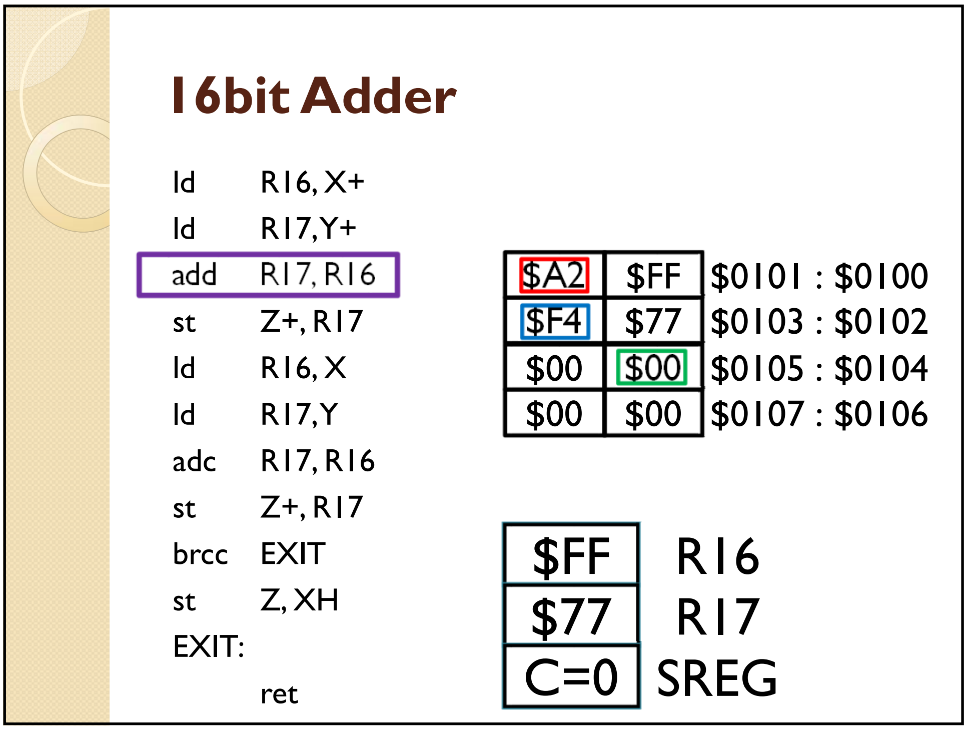


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$77	R17
C=0	SREG

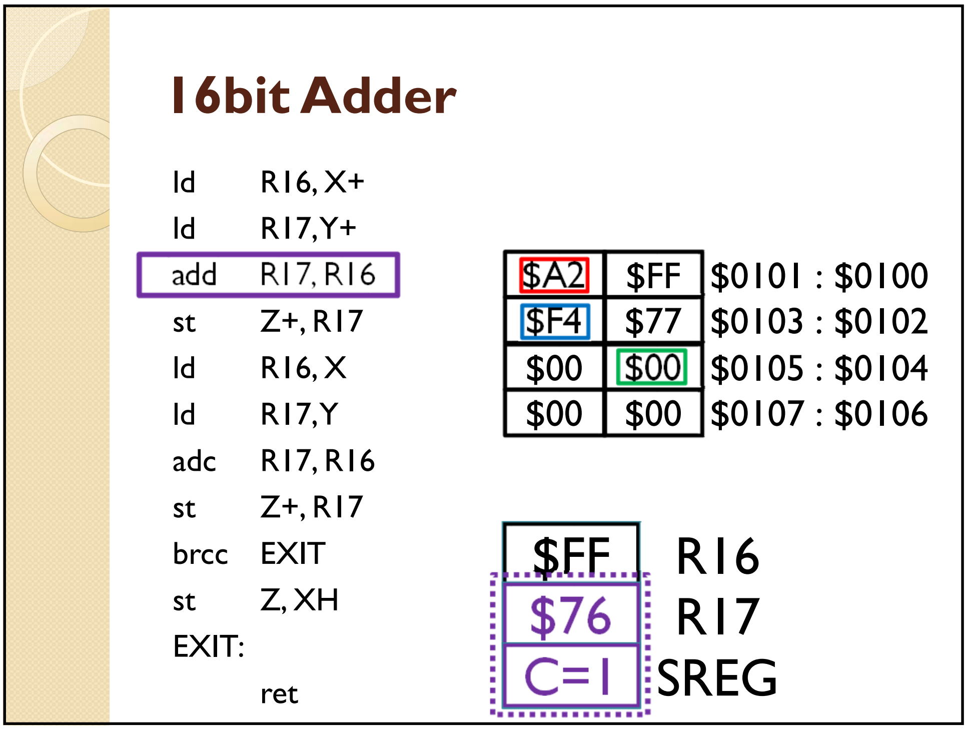


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$77	R17
C=0	SREG

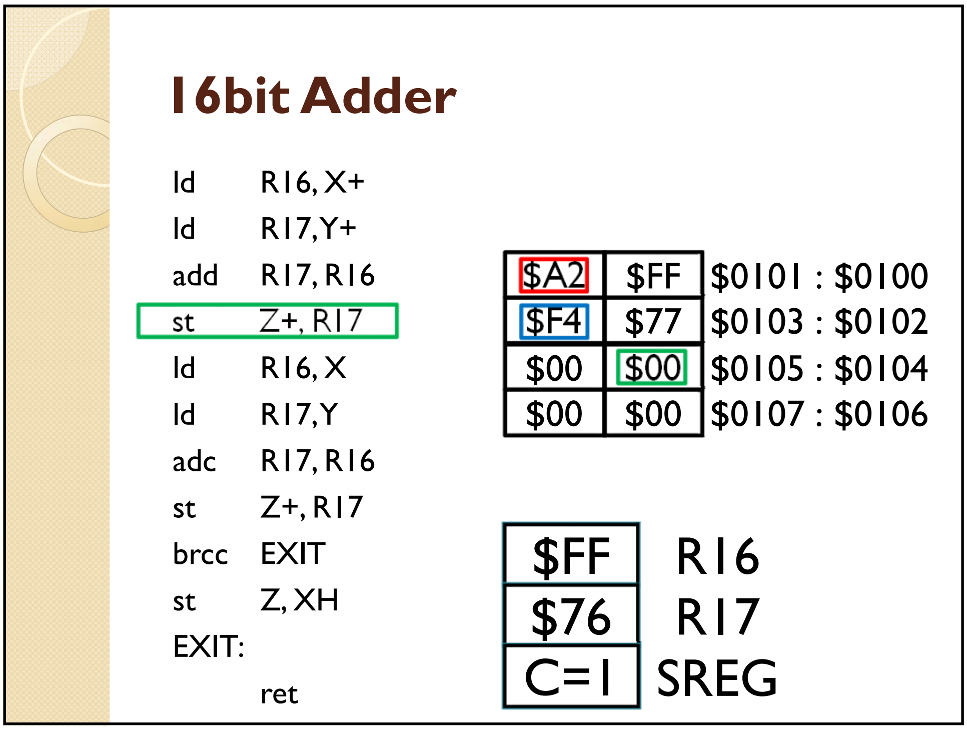


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG

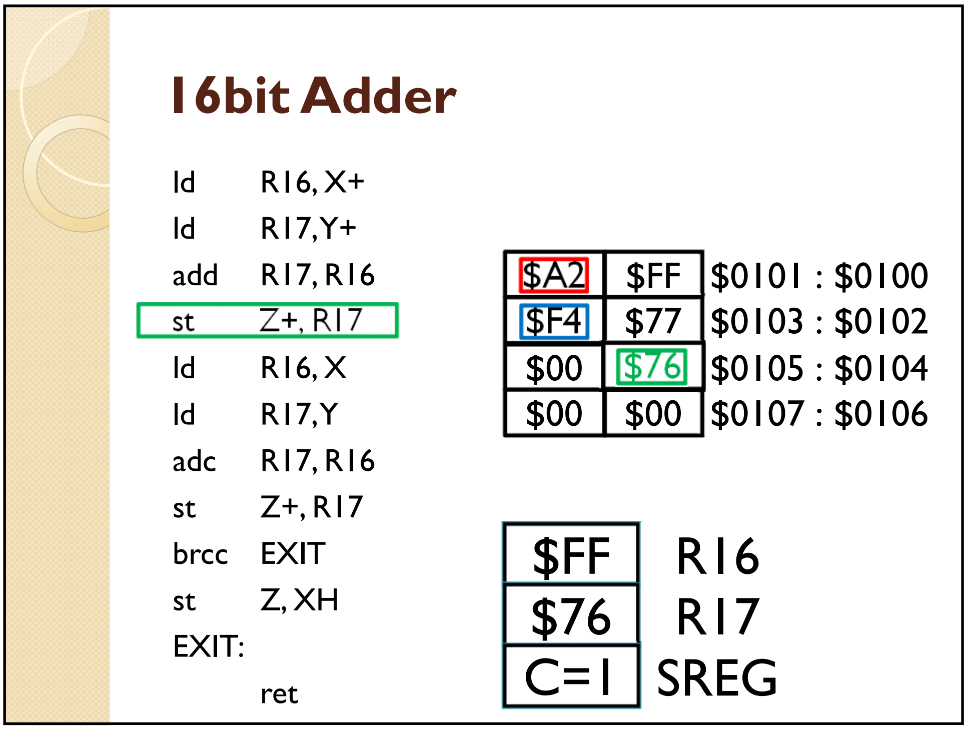


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$00	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG

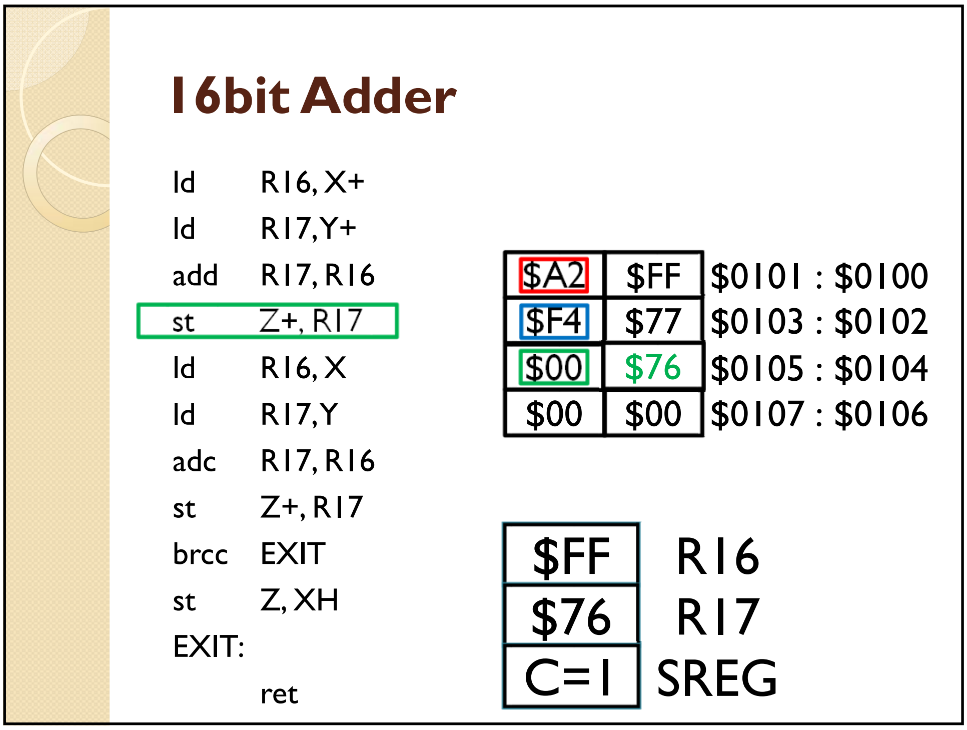


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$76	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG

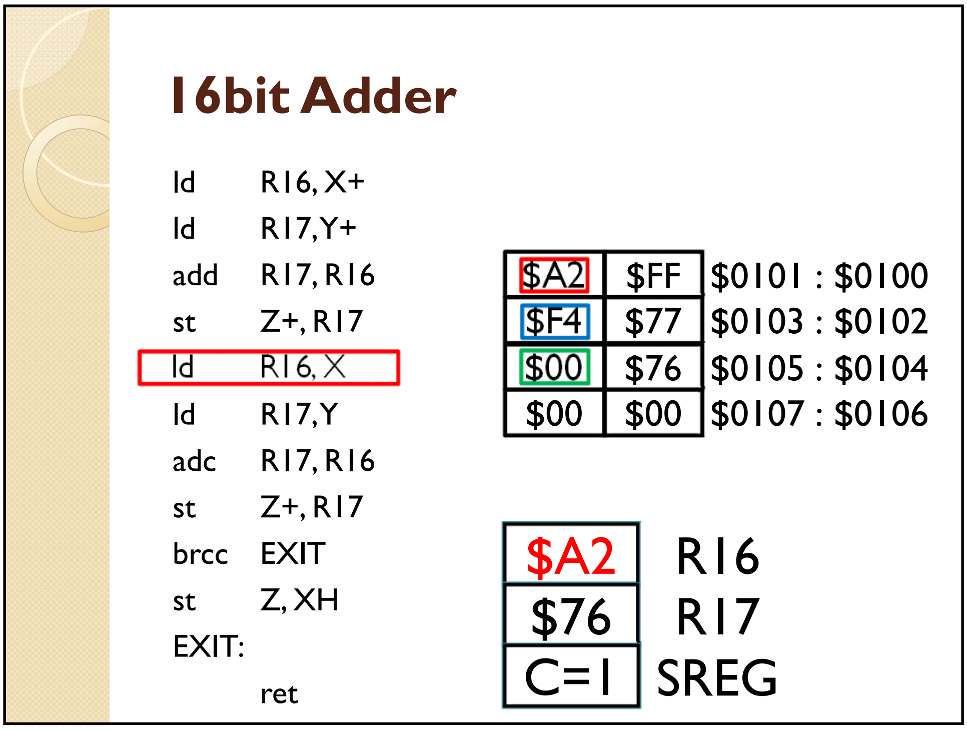


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$76	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG

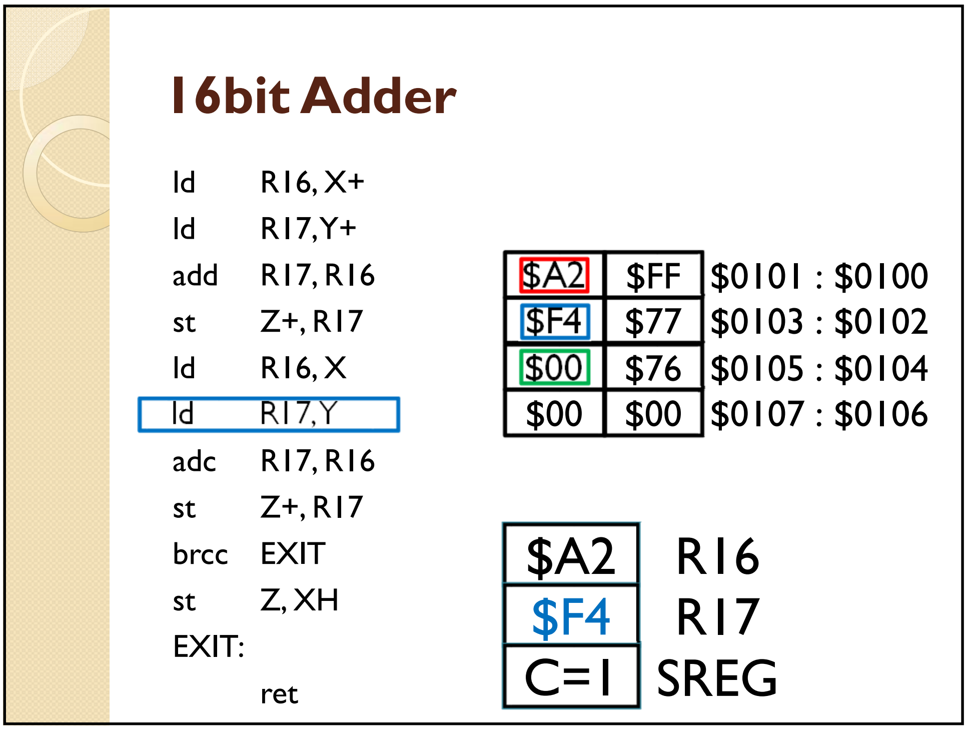


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$76	$0105 : $0104
$00	$00	$0107 : $0106

$A2	R16
$76	R17
C=1	SREG

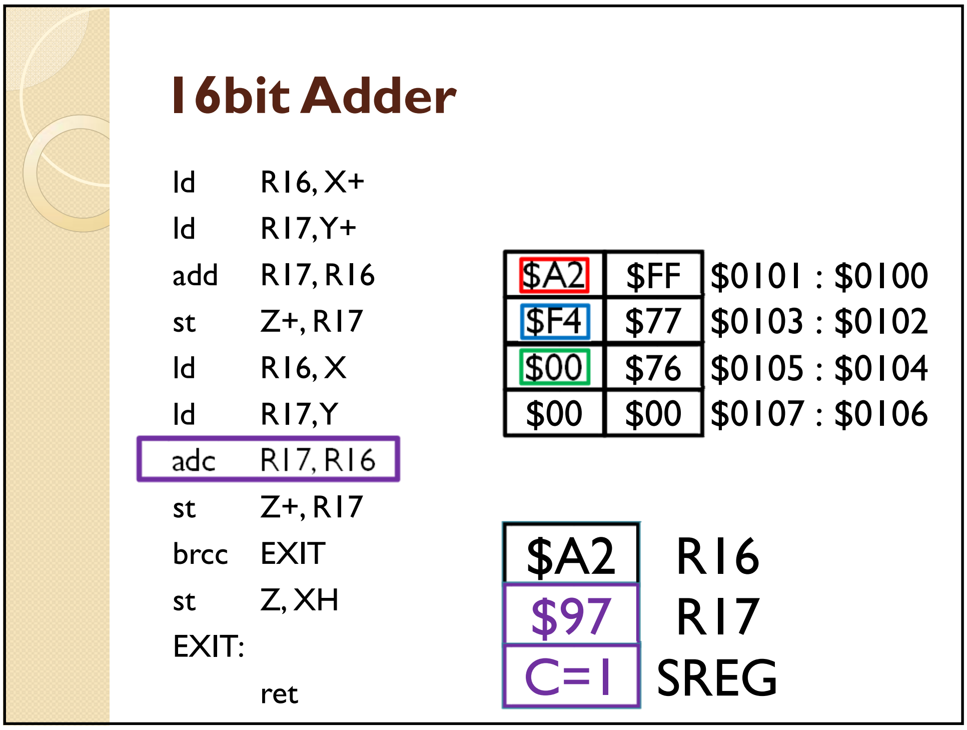


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$76	$0105 : $0104
$00	$00	$0107 : $0106

$A2	R16
$F4	R17
C=1	SREG

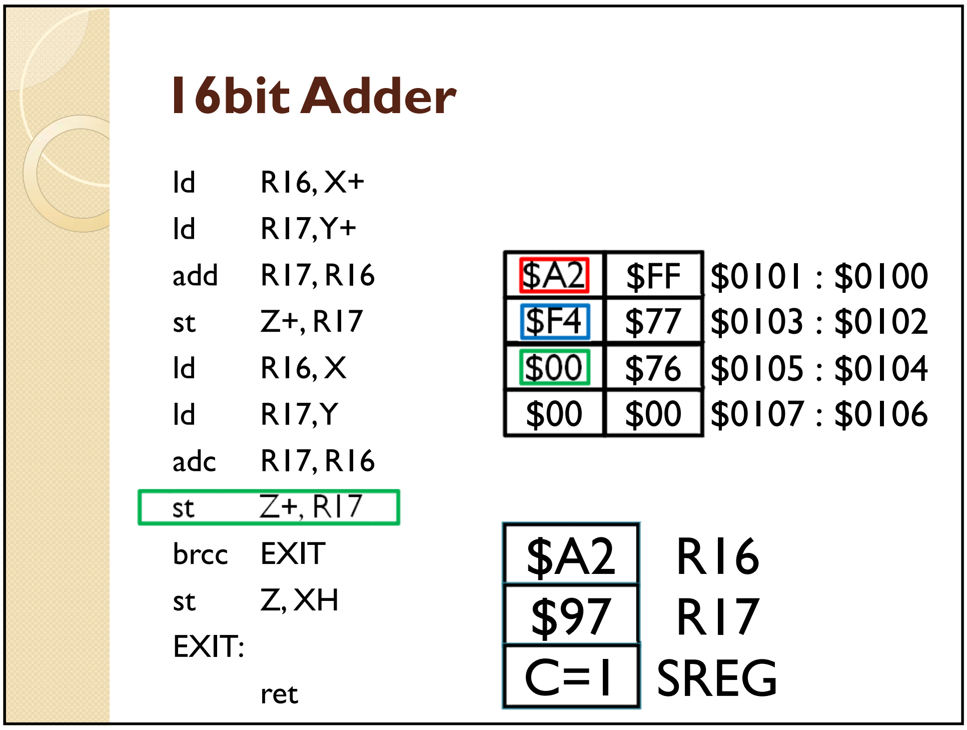


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$76	$0105 : $0104
$00	$00	$0107 : $0106

$A2	R16
$97	R17
C=1	SREG



16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$00	$76	$0105 : $0104
$00	$00	$0107 : $0106

$A2	R16
$97	R17
C=1	SREG

16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$97	$76	$0105 : $0104
$00	$00	$0107 : $0106

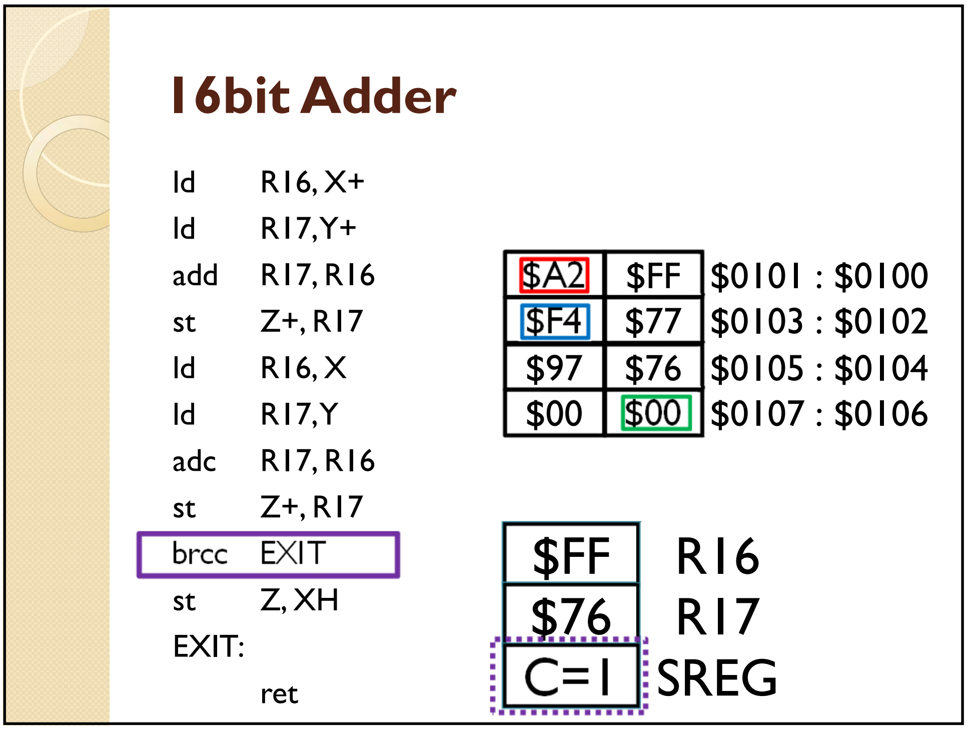
$A2	R16
$97	R17
C=1	SREG

16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$97	$76	$0105 : $0104
$00	$00	$0107 : $0106

$A2	R16
$97	R17
C=1	SREG

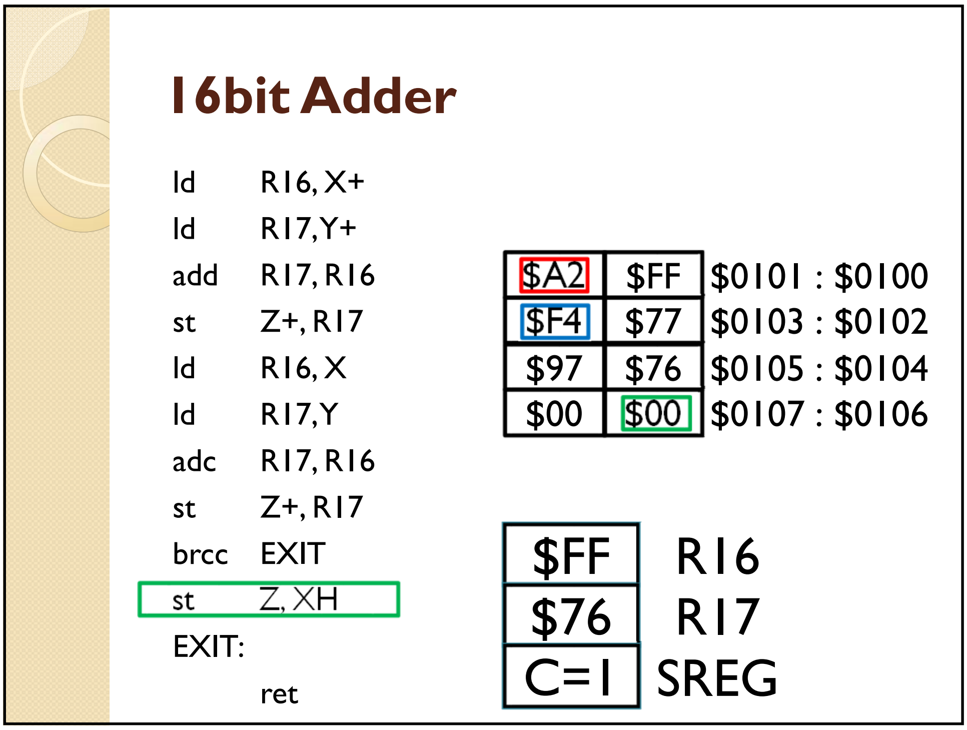


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$97	$76	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG

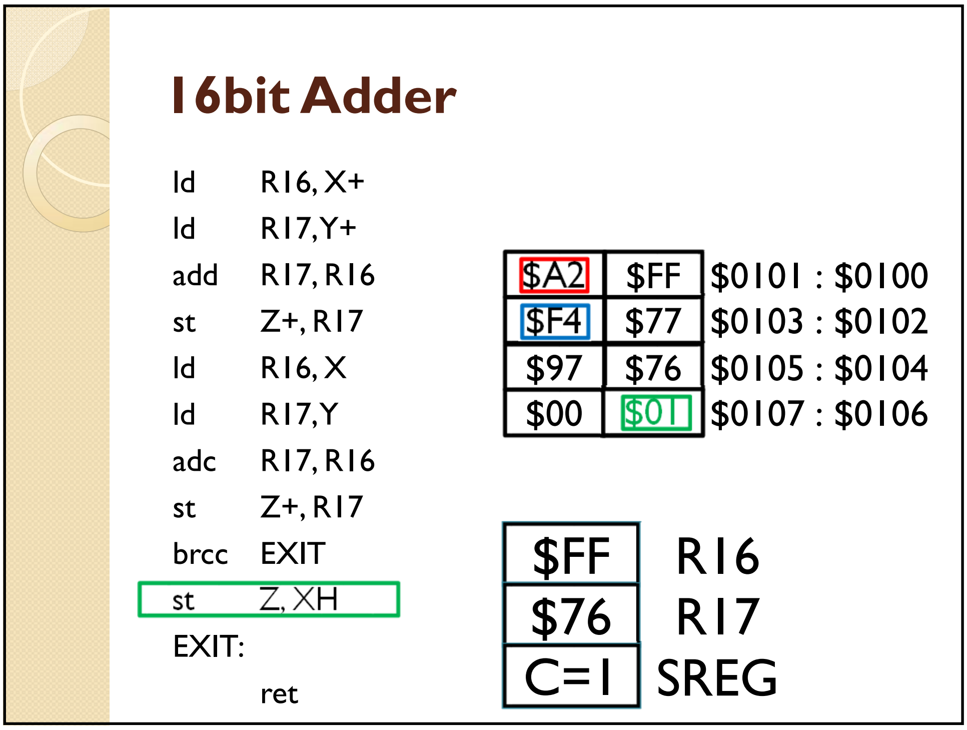


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$97	$76	$0105 : $0104
$00	$00	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG

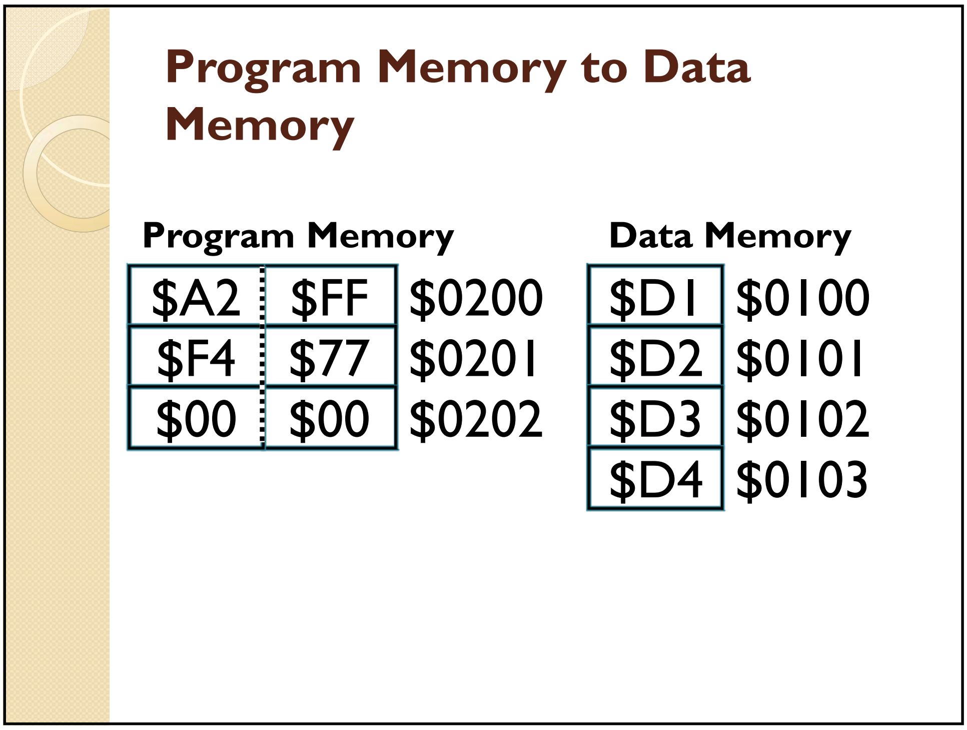


16bit Adder

```
ld      R16, X+
ld      R17,Y+
add     R17, R16
st      Z+, R17
ld      R16, X
ld      R17,Y
adc     R17, R16
st      Z+, R17
brcc    EXIT
st      Z, XH
EXIT:
        ret
```

$A2	$FF	$0101 : $0100
$F4	$77	$0103 : $0102
$97	$76	$0105 : $0104
$00	$01	$0107 : $0106

$FF	R16
$76	R17
C=1	SREG



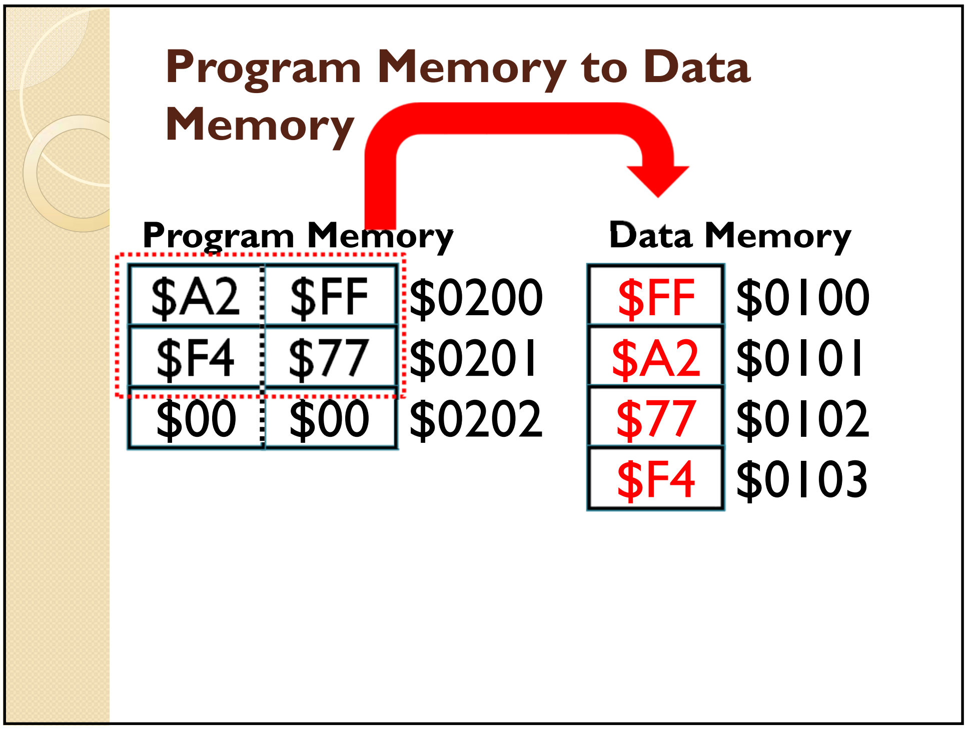
Program Memory to Data Memory

Program Memory

$A2	$FF	$0200
$F4	$77	$0201
$00	$00	$0202

Data Memory

$D1	$0100
$D2	$0101
$D3	$0102
$D4	$0103



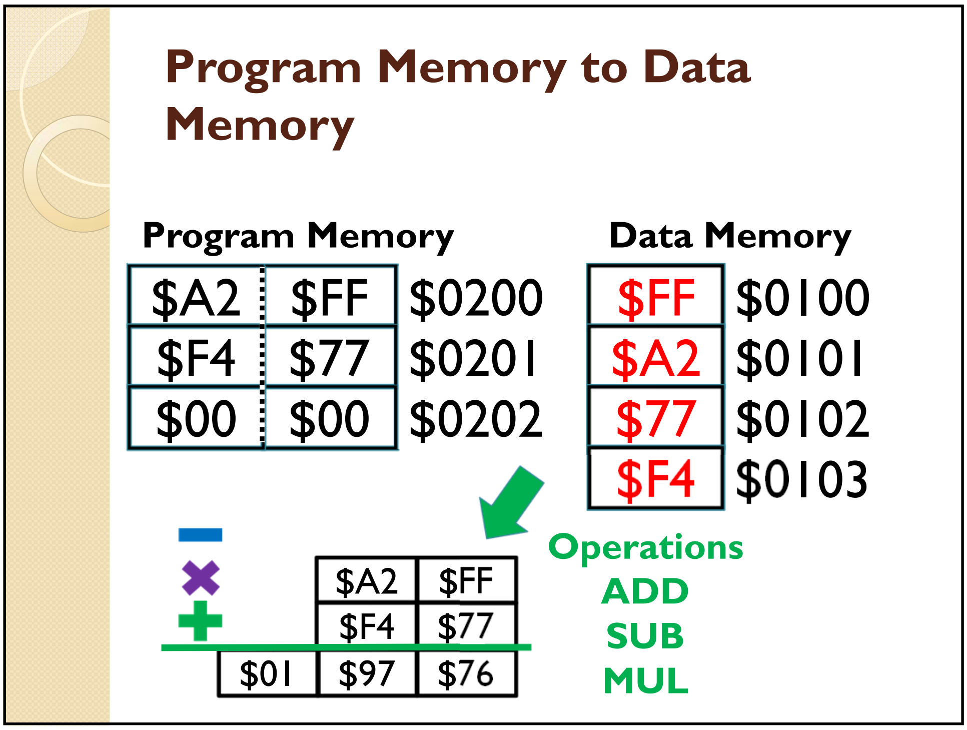
Program Memory to Data Memory

Program Memory

$A2	$FF	$0200
$F4	$77	$0201
$00	$00	$0202

Data Memory

$FF	$0100
$A2	$0101
$77	$0102
$F4	$0103



Program Memory to Data Memory

Program Memory

$A2	$FF	$0200
$F4	$77	$0201
$00	$00	$0202

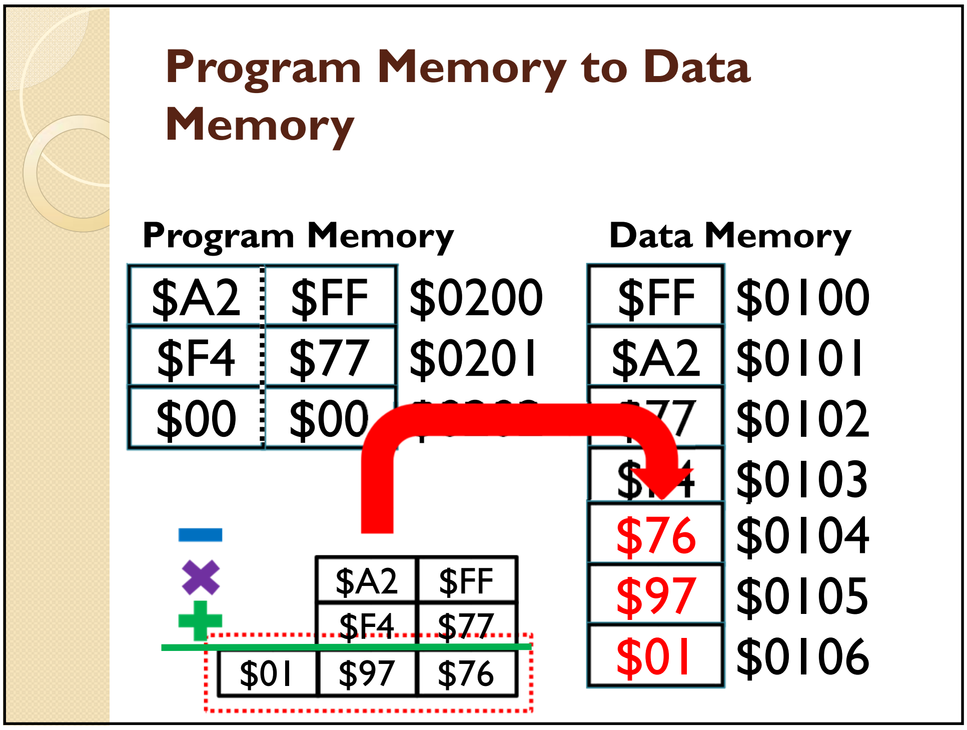
Data Memory

$FF	$0100
$A2	$0101
$77	$0102
$F4	$0103

−
×
+

	$A2	$FF
	$F4	$77
$01	$97	$76

Operations
ADD
SUB
MUL



Program Memory to Data Memory

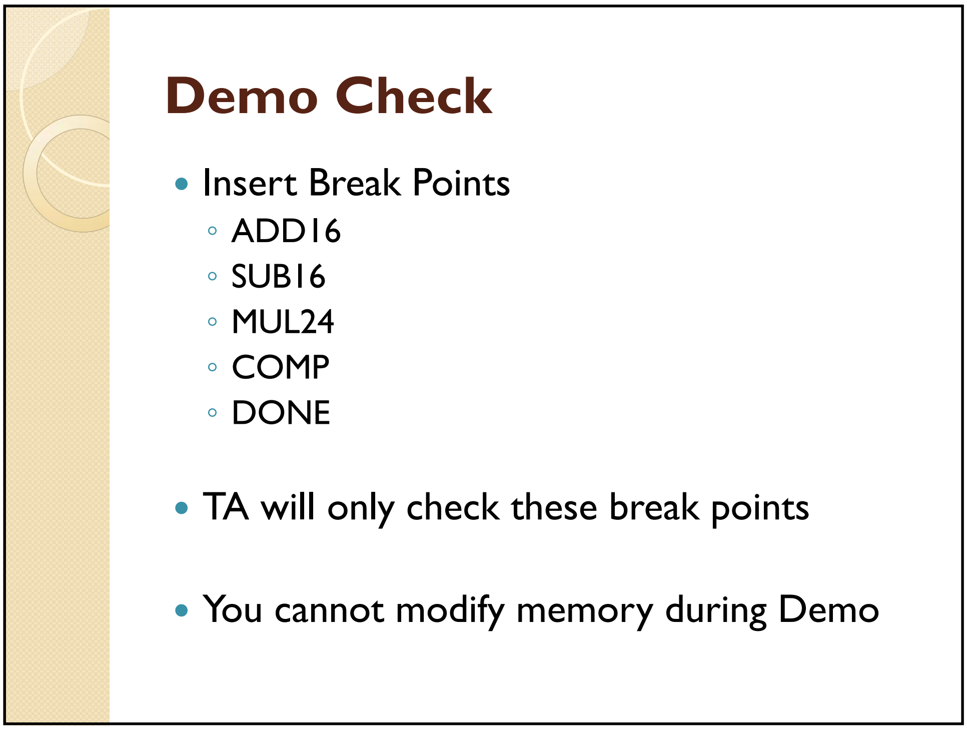
Program Memory

$A2	$FF	$0200
$F4	$77	$0201
$00	$00	$0202

Data Memory

$FF	$0100
$A2	$0101
$77	$0102
$F4	$0103
$76	$0104
$97	$0105
$01	$0106

	$A2	$FF
	$F4	$77
$01	$97	$76



Demo Check

- Insert Break Points
  - ADD16
  - SUB16
  - MUL24
  - COMP
  - DONE
- TA will only check these break points
- You cannot modify memory during Demo

Checklist for Lab 5

- Demo Checklist
  - Correct ADD16 result w/ direct operands
  - Correct SUB16 result w/ direct operands
  - Correct MUL24 result w/ direct operands
  - Correct ((D – E) + F)^2 result
  - Good explanation of changes required to make MUL16 into a working MUL24?
- Challenge Checklist
  - MUL24 implemented using shift-and-add, and used for ((D – E) + F)^2 calculation

Questions?