



**NTNU – Trondheim**  
Norwegian University of  
Science and Technology

## **Recitation lecture: problem set 6**

# 16 bytes alignment – why?

- Defined by widely used ABIs, your program will segfault if you don't follow it
- Boring answer, shouldn't really need to care
- SSE-instructions working on 16 byte vector registers

– Streamlined  
operator

`mulss xmm1, xmm0`

	127	95	63	31	0
XMM0	4.0	3.0	2.0	1.0	
				*	
XMM1	5.0	5.0	5.0	5.0	
				=	
XMM1	4.0	3.0	2.0	5.0	

`mulps xmm1, xmm0`

	127	95	63	31	0
	4.0	3.0	2.0	1.0	
	*	*	*	*	
	5.0	5.0	5.0	5.0	
	=	=	=	=	
	20.0	15.0	10.0	5.0	

utilize vector



# Intro to PS 6: Control structures

- Short assignment
  - Make our compiler generate interesting programs
- If you completed PS5, this one should be just the victory lap

# Intro to PS 6: If-statement

- If/else statement
- Compare with **cmp S2, S1** instruction, sets condition codes according to S1 - S2
- Conditional jump: **jne, je, jg, jge, jl, jle**
- No trouble having an **else** label just pointing at the **endif** label, if no else clause (Open for personal design preferences)

```
// compute a
// push a
// compute b
// pop a
cmp a, b
jne .ELSE
// if-block
jmp .ENDIF
.ELSE:
// opt. else-block
.ENDIF:
// continued program flow
```



# Intro to PS 6: While-statement

- While statement
- Much same as if-statement
- End of while block: return to start for a new computation of condition
- No loop optimizations expected

```
.WHILE:  
// compute a  
// push a  
// compute b  
// pop a  
cmp a, b  
jne .ENDWHILE  
// while-block  
jmp .WHILE  
.ENDWHILE:  
// continued program flow
```

# Intro to PS 6: Null-statement

- Null statement: skip an iteration of a loop
- Nested loops
  - Keep track of which to skip an iteration of
  - Innermost loop

```
while a < b
  while i < c
    continue // goto while i < c
```



# Compiler completed

- Upon completing this assignment, you have written a complete compiler from scratch
  - Scanner
  - Parser
  - Symbol table
  - Optimization (constant expression elimination)
  - Code generation