Modern C++ in an Embedded World

The Saga Continues

ciere consulting

Michael Caisse

CppCon 2018 | mcaisse@ciere.com | follow @MichaelCaisse

Contemporary C++ on Bare Metal Embedded

circa 2018

ciere consulting

Michael Caisse

CppCon 2018 | mcaisse@ciere.com | follow @MichaelCaisse

ciere.com

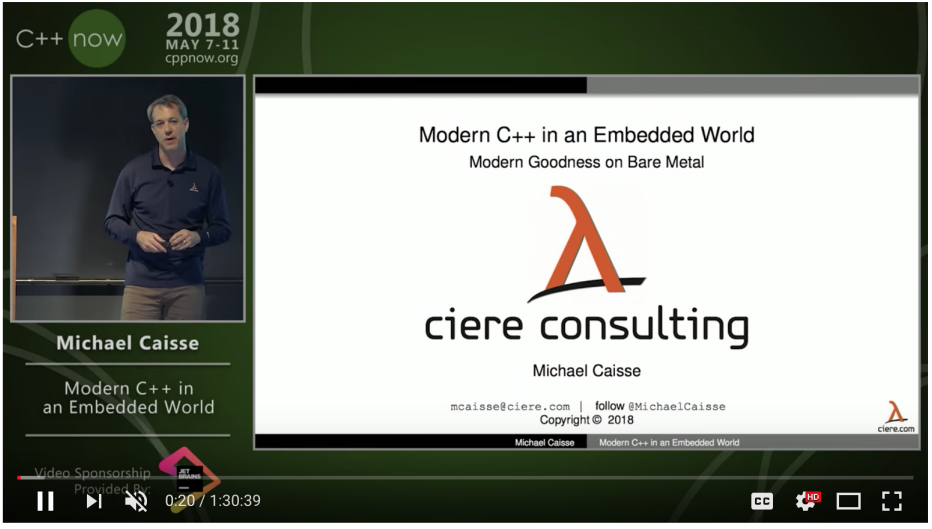
Modern C++ in an Embedded World

The Saga Continues

ciere consulting

Michael Caisse

CppCon 2018 | mcaisse@ciere.com | follow @MichaelCaisse



C++Now 2018: Michael Caisse “Modern C++ in Embedded Systems”

9,865 views

214 9 SHARE

Xavier Thomas 3 months ago (edited)

The real stuff starts at 26:30.
The good stuff starts at 42:40
You're welcome ;)

21 REPLY

Part I

The Project

VALVE MANIFOLD

PRESSURE

RETURN

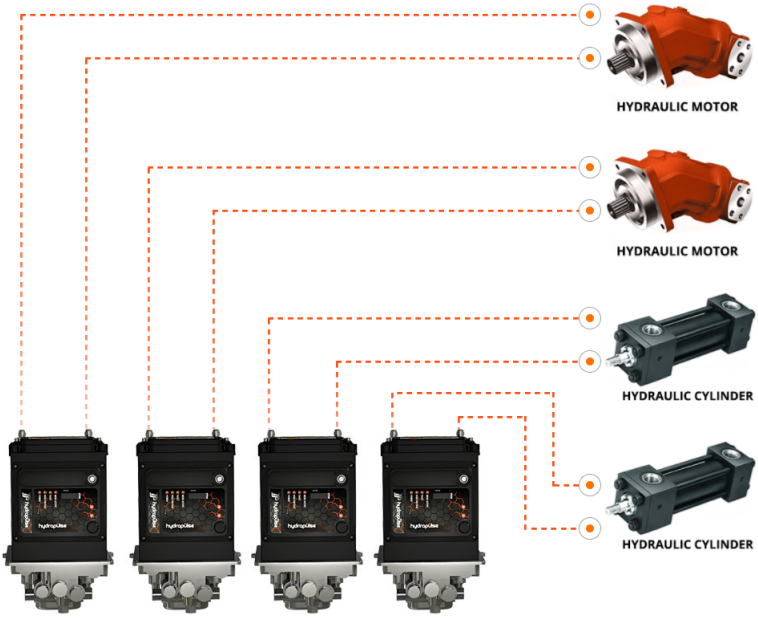
RETURN OIL COOLER

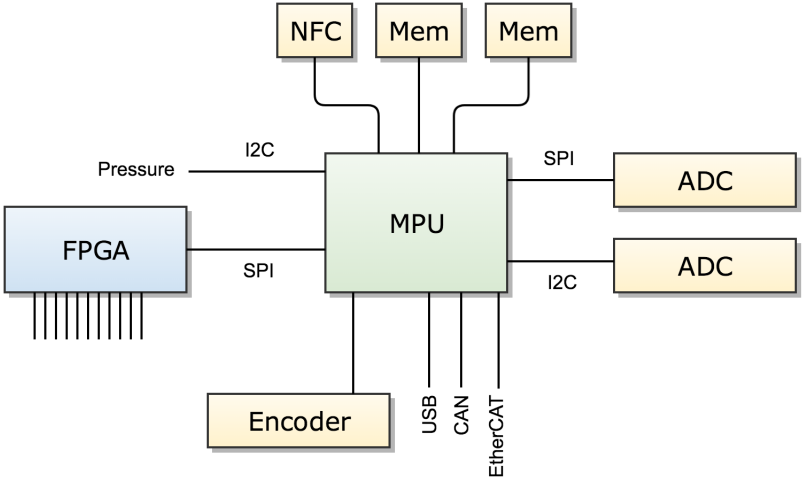
HYDRAULIC MOTOR

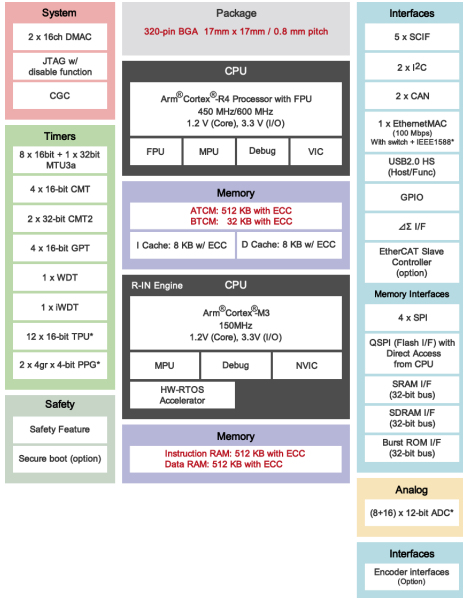
HYDRAULIC MOTOR

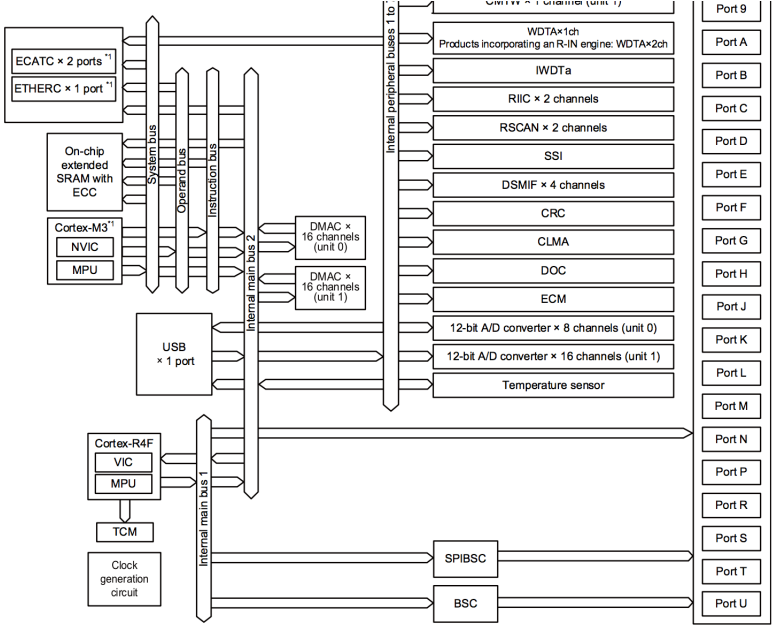
HYDRAULIC CYLINDER

HYDRAULIC CYLINDER

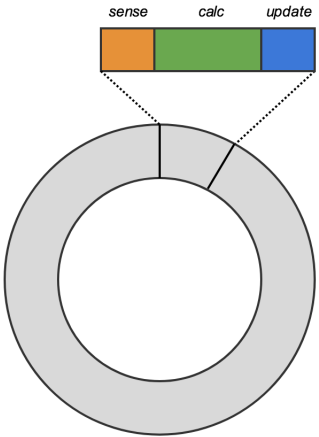

HYDRAULIC MOTOR
HYDRAULIC MOTOR
HYDRAULIC CYLINDER
HYDRAULIC CYLINDER


NFC
Mem
Mem
I2C
Pressure
SPI
ADC
FPGA
MPU
SPI
ADC
I2C
Encoder
USB
CAN
EtherCAT


System
2 x 16ch DMAC
JTAG w/ disable function
CGC
Timers
8 x 16bit + 1 x 32bit MTU3a
4 x 16-bit CMT
2 x 32-bit CMT2
4 x 16-bit GPT
1 x WDT
1 x IWDT
12 x 16-bit TPU*
2 x 4gr x 4-bit PPG*
Safety
Safety Feature
Secure boot (option)
Package
320-pin BGA 17mm x 17mm / 0.8 mm pitch
CPU
Arm® Cortex®-R4 Processor with FPU
450 MHz/600 MHz
1.2 V (Core), 3.3 V (I/O)
FPU
MPU
Debug
VIC
Memory
ATCM: 512 KB with ECC
BTCM: 32 KB with ECC
I Cache: 8 KB w/ ECC
D Cache: 8 KB w/ ECC
R-IN Engine
CPU
Arm® Cortex®-M3
150MHz
1.2V (Core), 3.3V (I/O)
MPU
Debug
NVIC
HW-RTOS Accelerator
Memory
Instruction RAM: 512 KB with ECC
Data RAM: 512 KB with ECC
Interfaces
5 x SCIF
2 x I2C
2 x CAN
1 x EthernetMAC
(100 Mbps)
With switch + IEEE1588*
USB2.0 HS (Host/Func)
GPIO
ΔΣ I/F
EtherCAT Slave Controller (option)
Memory Interfaces
4 x SPI
QSPI (Flash I/F) with Direct Access from CPU
SRAM I/F (32-bit bus)
SDRAM I/F (32-bit bus)
Burst ROM I/F (32-bit bus)
Analog
(8+16) x 12-bit ADC*
Interfaces
Encoder interfaces
(Option)


ECATC × 2 ports *1
ETHERC × 1 port *1
On-chip extended SRAM with ECC
Cortex-M3*1
NVIC
MPU
System bus
Operand bus
Instruction bus
Internal main bus 2
DMAC × 16 channels (unit 0)
DMAC × 16 channels (unit 1)
USB × 1 port
Internal peripheral buses 1 to
WDTA×1ch
Products incorporating an R-IN engine: WDTA×2ch
IWDTa
RIIC × 2 channels
RSCAN × 2 channels
SSI
DSMIF × 4 channels
CRC
CLMA
DOC
ECM
12-bit A/D converter × 8 channels (unit 0)
12-bit A/D converter × 16 channels (unit 1)
Temperature sensor
Port 9
Port A
Port B
Port C
Port D
Port E
Port F
Port G
Port H
Port J
Port K
Port L
Port M
Port N
Port P
Port R
Port S
Port T
Port U
Cortex-R4F
VIC
MPU
TCM
Clock generation circuit
Internal main bus 1
SPIBSC
BSC

Hard Real-Time



Why use C++ for this project?

Why C++

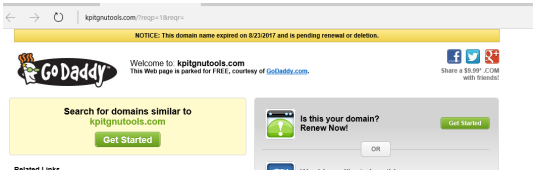
- I submitted a talk to C++Now about C++ and embedded
- C++ provides better abstractions
- Fewer bugs, finished sooner

Why C++

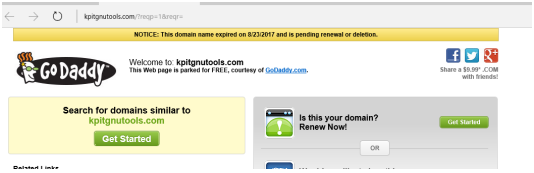
- I submitted a talk to C++Now about C++ and embedded
- C++ provides better abstractions
- Fewer bugs, finished sooner

ciere.com

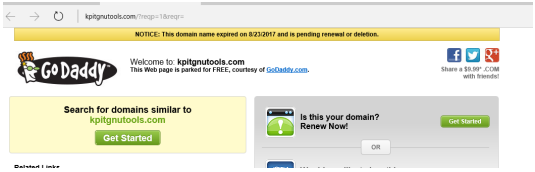
The Original Saga

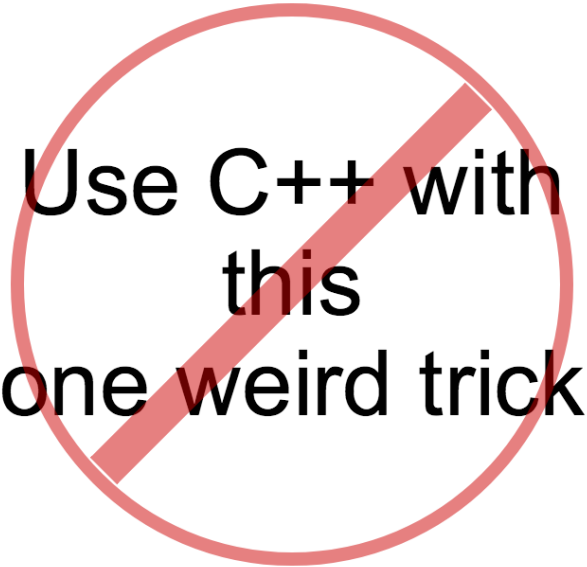


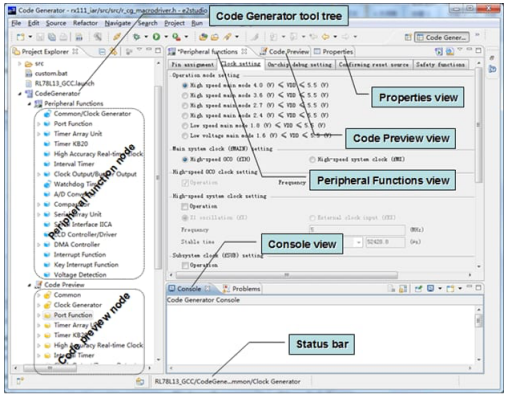
The Original Saga



The Original Saga



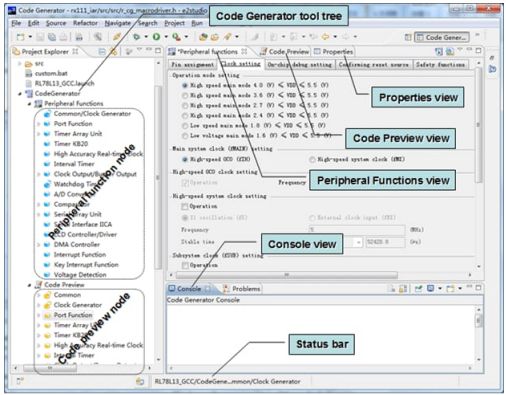

Use C++ with
this
one weird trick



T.T.H.W.

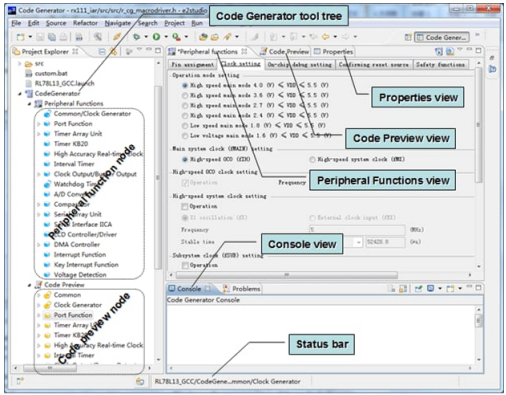
Time To Hello World

ciere.com



The good

- Easy to get development boards running
- Built-in target upload
- Debugging without hassle



The problems

- Centered around single-developer work flow
- Hard to integrate with a team
- Hard to integrate with a CI system
- Hides so much magic....

Standard Tools

Using standard tools:

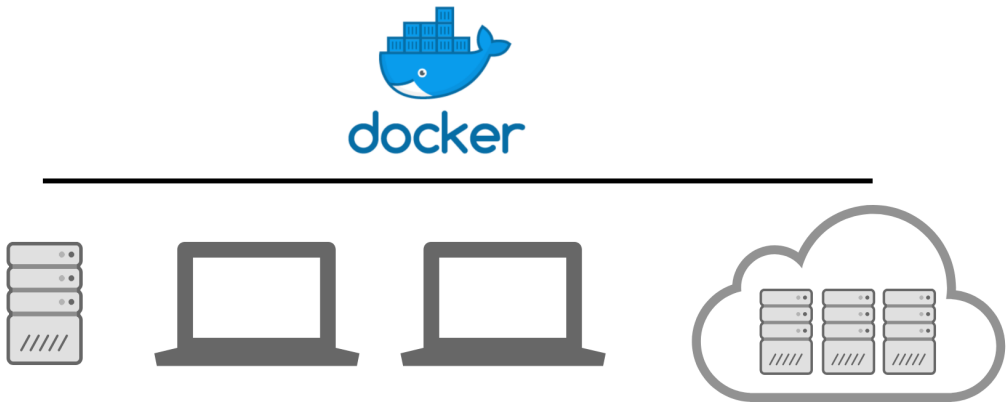
- GNU GCC from developer.arm.com
- CMake
- Static analysis and analyzers
- Continuous Integration
- Continuous Deployment

ciere.com

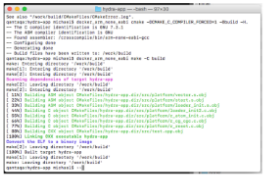
Part II

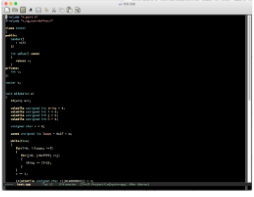
Making it all work

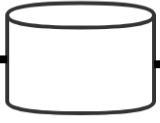
Heterogeneous Environment


docker

Dockcross







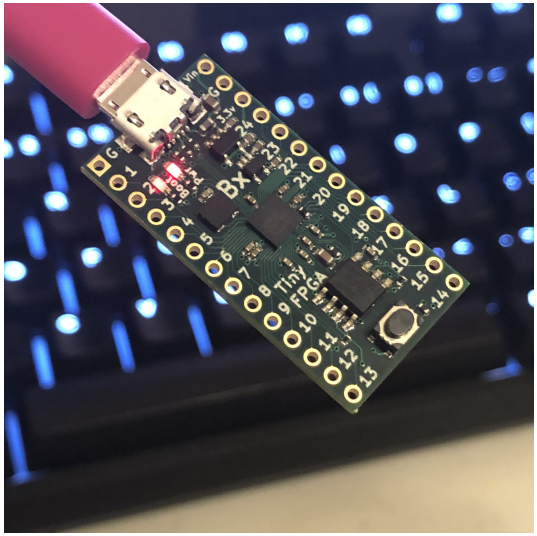
Dockcross

github : dockcross

Dockcross Everywhere

- TinyFPGA BX - Luke Valenty
- Open Hardware : ICE40LP8K
- 7,680 Logic Cells
- tinyfpga.com

- PicoRV32 - Clifford Wolf
- RISC-V processor IMC instruction set
- Open source tools : IceStorm



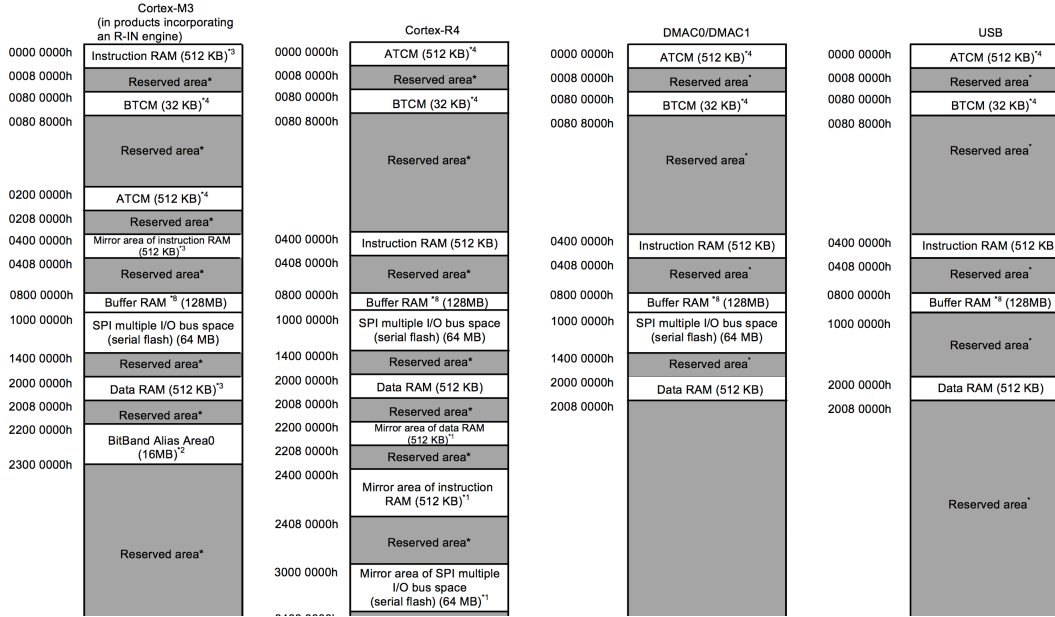
The Tools

Using standard tools:

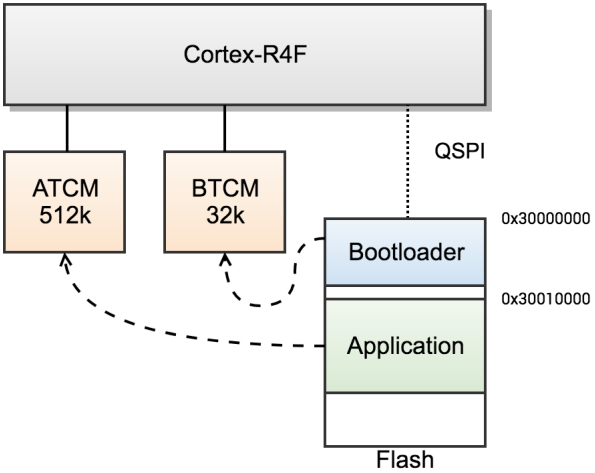
- GNU GCC from developer.arm.com
- CMake
- Static analysis and analyzers
- Continuous Integration
- Continuous Deployment

I have a compiler ... now what!

- reset operation
- memory layouts
- start-up needs
- peripherial needs
- language needs ASM, C, C++


Cortex-M3
(in products incorporating an R-IN engine)
0000 0000h
Instruction RAM (512 KB)*3
0008 0000h
Reserved area*
0080 0000h
BTCM (32 KB)*4
0080 8000h
Reserved area*
0200 0000h
ATCM (512 KB)*4
0208 0000h
Reserved area*
0400 0000h
Mirror area of instruction RAM (512 KB)*3
0408 0000h
Reserved area*
0800 0000h
Buffer RAM *8 (128MB)
1000 0000h
SPI multiple I/O bus space (serial flash) (64 MB)
1400 0000h
Reserved area*
2000 0000h
Data RAM (512 KB)*3
2008 0000h
Reserved area*
2200 0000h
BitBand Alias Area0 (16MB)*2
2300 0000h
Reserved area*
Cortex-R4
0000 0000h
ATCM (512 KB)*4
0008 0000h
Reserved area*
0080 0000h
BTCM (32 KB)*4
0080 8000h
Reserved area*
0400 0000h
Instruction RAM (512 KB)
0408 0000h
Reserved area*
0800 0000h
Buffer RAM *8 (128MB)
1000 0000h
SPI multiple I/O bus space (serial flash) (64 MB)
1400 0000h
Reserved area*
2000 0000h
Data RAM (512 KB)
2008 0000h
Reserved area*
2200 0000h
Mirror area of data RAM (512 KB)*1
2208 0000h
Reserved area*
2400 0000h
Mirror area of instruction RAM (512 KB)*1
2408 0000h
Reserved area*
3000 0000h
Mirror area of SPI multiple I/O bus space (serial flash) (64 MB)*1
DMAC0/DMAC1
0000 0000h
ATCM (512 KB)*4
0008 0000h
Reserved area*
0080 0000h
BTCM (32 KB)*4
0080 8000h
Reserved area*
0400 0000h
Instruction RAM (512 KB)
0408 0000h
Reserved area*
0800 0000h
Buffer RAM *8 (128MB)
1000 0000h
SPI multiple I/O bus space (serial flash) (64 MB)
1400 0000h
Reserved area*
2000 0000h
Data RAM (512 KB)
2008 0000h
USB
0000 0000h
ATCM (512 KB)*4
0008 0000h
Reserved area*
0080 0000h
BTCM (32 KB)*4
0080 8000h
Reserved area*
0400 0000h
Instruction RAM (512 KB)
0408 0000h
Reserved area*
0800 0000h
Buffer RAM *8 (128MB)
1000 0000h
Reserved area*
2000 0000h
Data RAM (512 KB)
2008 0000h
Reserved area*

Load to flash



EMBEDDED.fm

ESE101: SWITCHING FROM ASSEMBLY TO C

PART 3

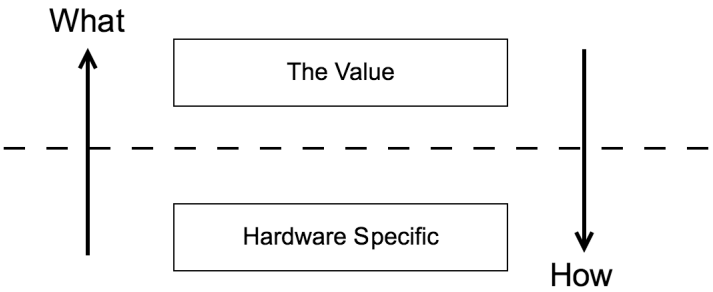
Part III

C++ Things

Why C++

- Zero cost abstraction
- Host debugging / implementation
- Correctness enforced by types
- Abstraction

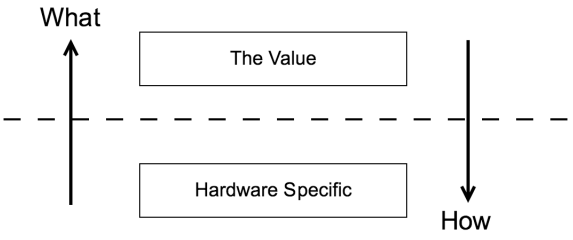
Why C++



Why C++

Watch Ben Deane's C++Now 2018 talk

"Easy to Use, Hard to Misuse: Declarative Style in C++"



Zero-cost abstractions

What is a zero-const abstraction?

Inline Example

```
int count_value(char ** input, int length, int value)
{
   int sum = 0;
   for(int i=0; i<length; ++i)
   {
      if(std::atoi(input[i]) == value)
      {
         ++sum;
      }
   }
   return sum;
}


int main(int argc, char** argv)
{
   // how many 8's
   return count_value(argv+1, argc-1, 8);
}
```

Inline Example

```
int count_value(char ** input, int length, int value)
{
   int sum = 0;
   for(int i=0; i<length; ++i)
   {
      if(std::atoi(input[i]) == value)
      {
         ++sum;
      }
   }
   return sum;
}


int main(int argc, char** argv)
{
   // how many 8's
   return count_value(argv+1, argc-1, 8);
}
```

Inline Example

```
count_value(char**, int, int):
        push    rbp
        push    r15
        push    r14
        push    rbx
        push    rax
        mov     r14d, edx
        mov     r15d, esi
        mov     rbp, rdi
        xor     ebx, ebx
        test    r15d, r15d
        jle     .LBB0_3
        xor     ebx, ebx

.LBB0_2:
        mov     rdi, qword ptr [rbp]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        xor     ecx, ecx
        cmp     eax, r14d
        sete    cl
        add     ebx, ecx
        add     rbp, 8
        dec     r15d
        jne     .LBB0_2

.LBB0_3:
        mov     eax, ebx
        add     rsp, 8
        pop     rbx
        pop     r14
        pop     r15
```

```
main:
        push    rbp
        push    r14
        push    rbx
        mov     rbx, rsi
        mov     ebp, edi
        xor     r14d, r14d
        test    ebp, ebp
        jle     .LBB1_3
        xor     r14d, r14d

.LBB1_2:
        mov     rdi, qword ptr [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        xor     ecx, ecx
        cmp     eax, 8
        sete    cl
        add     r14d, ecx
        add     rbx, 8
        dec     ebp
        jne     .LBB1_2

.LBB1_3:
        mov     eax, r14d
        pop     rbx
        pop     r14
        pop     rbp
        ret
```

Inline Example

```
count_value(char**, int, int):
        push    rbp
        push    r15
        push    r14
        push    rbx
        push    rax
        mov     r14d, edx
        mov     r15d, esi
        mov     rbp, rdi
        xor     ebx, ebx
        test    r15d, r15d
        jle     .LBB0_3
        xor     ebx, ebx

.LBB0_2:
        mov     rdi, qword ptr [rbp]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        xor     ecx, ecx
        cmp     eax, r14d
        sete    cl
        add     ebx, ecx
        add     rbp, 8
        dec     r15d
        jne     .LBB0_2

.LBB0_3:
        mov     eax, ebx
        add     rsp, 8
        pop     rbx
        pop     r14
        pop     r15
```

```
main:
        push    rbp
        push    r14
        push    rbx
        mov     rbx, rsi
        mov     ebp, edi
        xor     r14d, r14d
        test    ebp, ebp
        jle     .LBB1_3
        xor     r14d, r14d

.LBB1_2:
        mov     rdi, qword ptr [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        xor     ecx, ecx
        cmp     eax, 8
        sete    cl
        add     r14d, ecx
        add     rbx, 8
        dec     ebp
        jne     .LBB1_2

.LBB1_3:
        mov     eax, r14d
        pop     rbx
        pop     r14
        pop     rbp
        ret
```

Annonymous

```
namespace
{

   int count_value(char ** input, int length, int value)
   {
      int sum = 0;
      for(int i=0; i<length; ++i)
      {
         if(std::atoi(input[i]) == value)
         {
            ++sum;
         }
      }
      return sum;
   }

}


int main(int argc, char** argv)
{
  return count_value(argv+1, argc-1, 8);
}
```

Annonymous

```
namespace
{

   int count_value(char ** input, int length, int value)
   {
      int sum = 0;
      for(int i=0; i<length; ++i)
      {
         if(std::atoi(input[i]) == value)
         {
            ++sum;
         }
      }
      return sum;
   }

}


int main(int argc, char** argv)
{
  return count_value(argv+1, argc-1, 8);
}
```

Annonymous

```
main:
        push    rbp
        push    r14
        push    rbx
        mov     rbx, rsi
        mov     r14d, edi
        xor     ebp, ebp
        cmp     r14d, 2
        jl      .LBB0_3
        add     rbx, 8
        dec     r14d
        xor     ebp, ebp
.LBB0_2:
        mov     rdi, qword ptr [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        xor     ecx, ecx
        cmp     eax, 8
        sete    cl
        add     ebp, ecx
        add     rbx, 8
        dec     r14d
        jne     .LBB0_2
.LBB0_3:
        mov     eax, ebp
        pop     rbx
        pop     r14
        pop     rbp
        ret
```

Annonymous

Does this code bother you?

```
namespace
{

   int count_value(char ** input, int length, int value)
   {
      int sum = 0;
      for(int i=0; i<length; ++i)
      {
         if(std::atoi(input[i]) == value)
         {
            ++sum;
         }
      }
      return sum;
   }

}


int main(int argc, char** argv)
{
  return count_value(argv+1, argc-1, 8);
}
```

Annonymous

Does this code bother you?

```
namespace
{
   int count_value(char ** input, int length, int value)
   {
      int sum = 0;
      for(int i=0; i<length; ++i)
      {
         if(std::atoi(input[i]) == value)
         {
            ++sum;
         }
      }
      return sum;
   }
}

int main(int argc, char** argv)
{
  return count_value(argv+1, argc-1, 8);
}
```

Algorithms

```cpp
#include <algorithm>

int main(int argc, char ** argv)
{
    return
      std::count_if( argv+1, argv+argc,
                     [](auto arg)
                     {
                       return std::atoi(arg) == 8;
                     }
                 );
}
```

Algorithms

Raw Loop

```
main:
        push    rbp
        push    r14
        push    rbx
        mov     rbx, rsi
        mov     r14d, edi
        xor     ebp, ebp
        cmp     r14d, 2
        jl      .LBB0_3
        add     rbx, 8
        dec     r14d
        xor     ebp, ebp
.LBB0_2:
        mov     rdi, qword ptr [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        xor     ecx, ecx
        cmp     eax, 8
        sete    cl
        add     ebp, ecx
        add     rbx, 8
        dec     r14d
        jne     .LBB0_2
.LBB0_3:
        mov     eax, ebp
        pop     rbx
        pop     r14
        pop     rbp
        ret
```

Algorithm

```
main:
        movsx   rdi, edi
        push    r12
        push    rbp
        lea     r12, [rsi+rdi*8]
        push    rbx
        lea     rbx, [rsi+8]
        xor     ebp, ebp
        cmp     r12, rbx
        je      .L2
.L4:
        mov     rdi, QWORD PTR [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        cmp     eax, 8
        sete    al
        add     rbx, 8
        movzx   eax, al
        add     rbp, rax
        cmp     r12, rbx
        jne     .L4
.L2:
        mov     eax, ebp
        pop     rbx
        pop     rbp
        pop     r12
        ret
```

Algorithm - with Capture

```cpp
#include <algorithm>

int main(int argc, char ** argv)
{
  int value = 8;

  return
      std::count_if(argv + 1, argv + argc,
                    [value](auto arg)
                    {
                      return std::atoi(arg) == value;
                    }
                );
}
```

Algorithms

Without Capture

```
main:
        movsx   rdi, edi
        push    r12
        push    rbp
        lea     r12, [rsi+rdi*8]
        push    rbx
        lea     rbx, [rsi+8]
        xor     ebp, ebp
        cmp     r12, rbx
        je      .L2
.L4:
        mov     rdi, QWORD PTR [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        cmp     eax, 8
        sete    al
        add     rbx, 8
        movzx   eax, al
        add     rbp, rax
        cmp     r12, rbx
        jne     .L4
.L2:
        mov     eax, ebp
        pop     rbx
        pop     rbp
        pop     r12
        ret
```

With Capture

```
main:
        movsx   rdi, edi
        push    r12
        push    rbp
        lea     r12, [rsi+rdi*8]
        push    rbx
        lea     rbx, [rsi+8]
        xor     ebp, ebp
        cmp     r12, rbx
        je      .L2
.L4:
        mov     rdi, QWORD PTR [rbx]
        xor     esi, esi
        mov     edx, 10
        call    strtol
        cmp     eax, 8
        sete    al
        add     rbx, 8
        movzx   eax, al
        add     rbp, rax
        cmp     r12, rbx
        jne     .L4
.L2:
        mov     eax, ebp
        pop     rbx
        pop     rbp
        pop     r12
        ret
```

Counting Data

```
#include <algorithm>

int data[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
              9, 8, 7, 6, 5, 4, 3, 2, 1 };

int main()
{
   return
     std::count_if(data, data+sizeof(data),
                   [](auto v)
                   {
                     return v == 8;
                   }
             );
}
```

Counting Data

Thank you Jens Schmidt

```
#include <algorithm>

int data[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
              9, 8, 7, 6, 5, 4, 3, 2, 1 };

int main()
{
   return
     std::count_if(data, data+sizeof(data),
                   [](auto v)
                   {
                     return v == 8;
                   }
             );
}
```

Counting Data

```
#include <algorithm>
#include <iterator>

int data[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
              9, 8, 7, 6, 5, 4, 3, 2, 1 };

int main()
{
   return
     std::count_if(std::begin(data), std::end(data),
                   [](auto v)
                   {
                     return v == 8;
                   }
             );
}
```

Counting Data

Using `-Os` optimization:

```
main:
  mov rcx, -76
  xor eax, eax
.LBB0_1:
  xor edx, edx
  cmp dword ptr [rcx + data+76], 8
  sete dl
  add rax, rdx
  add rcx, 4
  jne .LBB0_1
  ret
```

```
data:
  .long 1 # 0x1
  .long 2 # 0x2
  .long 3 # 0x3
  .long 4 # 0x4
  .long 5 # 0x5
  .long 6 # 0x6
  .long 7 # 0x7
  .long 8 # 0x8
  .long 9 # 0x9
  .long 10 # 0xa
  .long 9 # 0x9
  .long 8 # 0x8
  .long 7 # 0x7
  .long 6 # 0x6
  .long 5 # 0x5
  .long 4 # 0x4
  .long 3 # 0x3
  .long 2 # 0x2
  .long 1 # 0x1
```

Importance of Being Const

```
int data[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
              9, 8, 7, 6, 5, 4, 3, 2, 1};

int main()
{
   return
     std::count_if(std::begin(data), std::end(data),
                   [](auto v)
                   {
                     return v == 8;
                   }
             );
}
```

Importance of Being Const

```
int const data[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
                    9, 8, 7, 6, 5, 4, 3, 2, 1};

int main()
{
   return
     std::count_if(std::begin(data), std::end(data),
                   [](auto v)
                   {
                     return v == 8;
                   }
             );
}
```

Importance of Being Const

```
int const data[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
                    9, 8, 7, 6, 5, 4, 3, 2, 1};

int main()
{
   return
     std::count_if(std::begin(data), std::end(data),
                   [](auto v)
                   {
                     return v == 8;
                   }
             );
}
```

```
main:
        mov     eax, 2
        ret
```

Polymorphism

What is polymorphism?

Runtime Polymorphism

```
int main(int argc, char** argv)
{
   simple s;
   float result = 0;

   for(int i=0; i<10000; ++i)
   {
      result += s.get_point(i);
   }

   return result;
}
```

Runtime Polymorphism

```
class curve
{
public:
   virtual float adjust(float) = 0;

   virtual float get_point(float x)
   {
      return adjust(x) * x;
   }
};


class simple : public curve
{
public:
   virtual float adjust(float x) override
   {
      return x*0.8;
   }
};
```

Runtime Polymorphism

```
class curve
{
public:
   virtual float adjust(float) = 0;

   virtual float get_point(float x)
   {
      return adjust(x) * x;
   }
};


class simple : public curve
{
public:
   virtual float adjust(float x) override
   {
      return x*0.8;
   }
};
```

Runtime Polymorphism

```
class curve
{
public:
   virtual float adjust(float) = 0;

   virtual float get_point(float x)
   {
      return adjust(x) * x;
   }
};


class simple : public curve
{
public:
   virtual float adjust(float x) override
   {
      return x*0.8;
   }
};
```

Runtime Polymorphism

```
class curve
{
public:
   virtual float adjust(float) = 0;

   virtual float get_point(float x)
   {
      return adjust(x) * x;
   }
};

class simple : public curve
{
public:
   virtual float adjust(float x) override
   {
      return x*0.8;
   }
};
```

Runtime Polymorphism

```
class curve
{
public:
   virtual float adjust(float) = 0;

   virtual float get_point(float x)
   {
      return adjust(x) * x;
   }
};

class simple : public curve
{
public:
   virtual float adjust(float x) override
   {
      return x*0.8;
   }
};
```

Runtime Polymorphism

```
curve::get_point(float):
        sub     rsp, 24
        mov     rax, QWORD PTR [rdi]
        movss   DWORD PTR [rsp+12], xmm0
        call    [QWORD PTR [rax]]
        movss   xmm1, DWORD PTR [rsp+12]
        add     rsp, 24
        mulss   xmm0, xmm1
        ret
simple::adjust(float):
        cvtss2sd        xmm0, xmm0
        mulsd   xmm0, QWORD PTR .LC0[rip]
        cvtsd2ss        xmm0, xmm0
        ret
main:
        push    rbx
        pxor    xmm0, xmm0
        mov     eax, OFFSET FLAT:simple::adjust(float)
        xor     ebx, ebx
        sub     rsp, 32
        mov     QWORD PTR [rsp+16], OFFSET FLAT:vtable for simple+16
        movss   DWORD PTR [rsp+24], xmm0
        movss   DWORD PTR [rsp+8], xmm0
        jmp     .L6
.L9:
        mov     rax, QWORD PTR [rsp+16]
        mov     rax, QWORD PTR [rax]
.L6:
        pxor    xmm1, xmm1
        lea     rdi, [rsp+16]
        cvtsi2ss        xmm1, ebx
        add     ebx, 1
        movaps  xmm0, xmm1
        movss   DWORD PTR [rsp+12], xmm1
        call    rax
        movss   xmm1, DWORD PTR [rsp+12]
        cmp     ebx, 10000
        mulss   xmm1, xmm0
        addss   xmm1, DWORD PTR [rsp+8]
        movss   DWORD PTR [rsp+8], xmm1
        jne     .L9
        cvttss2si       eax, DWORD PTR [rsp+8]
        add     rsp, 32
        pop     rbx
        ret
typeinfo name for curve:
        .string "5curve"
typeinfo for curve:
        .quad   vtable for __cxxabiv1::__class_type_info+16
        .quad   typeinfo name for curve
typeinfo name for simple:
        .string "6simple"
typeinfo for simple:
        .quad   vtable for __cxxabiv1::__si_class_type_info+1
        .quad   typeinfo name for simple
        .quad   typeinfo for curve
vtable for simple:
        .quad   0
        .quad   typeinfo for simple
        .quad   simple::adjust(float)
        .quad   curve::get_point(float)
.LC0:
        .long   2576980378
        .long   1072273817
```

Static Polymorphism

```
int main(int argc, char** argv)
{
  simple s;
  float result = 0;

  for(int i=0; i<10000; ++i)
  {
    result += s.get_point(i);
  }

  return result;
}
```

Static Polymorphism - CRTP

```
template <typename D>
class curve
{
  public:
      float get_point(float x)
      {
        return impl().adjust(x) * x;
      }
  private:
      D& impl()
      {
        return * static_cast<D*>(this);
      }
};


class simple : public curve<simple>
{
  public:
      float adjust(float x)
      {
         return x*0.8;
      }
};
```

Static Polymorphism - CRTP

```
template <typename D>
class curve
{
  public:
      float get_point(float x)
      {
        return impl().adjust(x) * x;
      }
  private:
      D& impl()
      {
        return * static_cast<D*>(this);
      }
};


class simple : public curve<simple>
{
  public:
      float adjust(float x)
      {
         return x*0.8;
      }
};
```

Static Polymorphism - CRTP

```
template <typename D>
class curve
{
  public:
      float get_point(float x)
      {
        return impl().adjust(x) * x;
      }
  private:
      D& impl()
      {
        return * static_cast<D*>(this);
      }
};


class simple : public curve<simple>
{
  public:
      float adjust(float x)
      {
         return x*0.8;
      }
};
```

Static Polymorphism - CRTP

```
template <typename D>
class curve
{
  public:

     float get_point(float x)
     {
       return impl().adjust(x) * x;
     }

  private:

     D& impl()
     {
       return * static_cast<D*>(this);
     }

};


class simple : public curve<simple>
{
  public:

     float adjust(float x)
     {
        return x*0.8;
     }

};
```

Static Polymorphism - CRTP

```
template <typename D>
class curve
{
  public:
      float get_point(float x)
      {
        return impl().adjust(x) * x;
      }
  private:
      D& impl()
      {
        return * static_cast<D*>(this);
      }
};


class simple : public curve<simple>
{
  public:
      float adjust(float x)
      {
         return x*0.8;
      }
};
```

Static Polymorphism - CRTP

```
main:
        pxor    xmm2, xmm2
        xor     eax, eax
        movsd   xmm3, QWORD PTR .LC1[rip]
.L2:
        pxor    xmm1, xmm1
        pxor    xmm0, xmm0
        cvtsi2ss        xmm1, eax
        add     eax, 1
        cmp     eax, 10000
        cvtss2sd        xmm0, xmm1
        mulsd   xmm0, xmm3
        cvtsd2ss        xmm0, xmm0
        mulss   xmm0, xmm1
        addss   xmm2, xmm0
        jne     .L2
        cvttss2si       eax, xmm2
        ret
.LC1:
        .long   2576980378
        .long   1072273817
```

Devirtualization

```
class curve
{
  public:
     virtual float adjust(float) = 0;

     virtual float get_point(float x)
     {
       return adjust(x) * x;
     }
};

class simple : public curve
{
  public:
     virtual float adjust(float x)
     {
        return x*0.8;
     }
};

int main(int argc, char** argv)
{
  simple s;
  float result = 0;

  for(int i=0; i<10000; ++i)
  {
    result += s.get_point(i);
  }
```

Devirtualization

```
main:
        pxor    xmm2, xmm2
        xor     eax, eax
        movsd   xmm3, QWORD PTR .LC1[rip]
.L2:
        pxor    xmm1, xmm1
        pxor    xmm0, xmm0
        cvtsi2ss        xmm1, eax
        add     eax, 1
        cmp     eax, 10000
        cvtss2sd        xmm0, xmm1
        mulsd   xmm0, xmm3
        cvtsd2ss        xmm0, xmm0
        mulss   xmm0, xmm1
        addss   xmm2, xmm0
        jne     .L2
        cvttss2si       eax, xmm2
        ret
.LC1:
        .long   2576980378
        .long   1072273817
```

Devirtualization

CRTP

```
main:
        pxor    xmm2, xmm2
        xor     eax, eax
        movsd   xmm3, QWORD PTR .LC1[rip]
.L2:
        pxor    xmm1, xmm1
        pxor    xmm0, xmm0
        cvtsi2ss        xmm1, eax
        add     eax, 1
        cmp     eax, 10000
        cvtss2sd        xmm0, xmm1
        mulsd   xmm0, xmm3
        cvtsd2ss        xmm0, xmm0
        mulss   xmm0, xmm1
        addss   xmm2, xmm0
        jne     .L2
        cvttss2si       eax, xmm2
        ret
.LC1:
        .long   2576980378
        .long   1072273817
```

Devirtualized

```
main:
        pxor    xmm2, xmm2
        xor     eax, eax
        movsd   xmm3, QWORD PTR .LC1[rip]
.L2:
        pxor    xmm1, xmm1
        pxor    xmm0, xmm0
        cvtsi2ss        xmm1, eax
        add     eax, 1
        cmp     eax, 10000
        cvtss2sd        xmm0, xmm1
        mulsd   xmm0, xmm3
        cvtsd2ss        xmm0, xmm0
        mulss   xmm0, xmm1
        addss   xmm2, xmm0
        jne     .L2
        cvttss2si       eax, xmm2
        ret
.LC1:
        .long   2576980378
        .long   1072273817
```

Zero Cost - Where's the Code?!

- Inlined functions don't look like the code we wrote
- `const` can allow the compiler to optimize LOCs away
- Static polymorphism and devirtualization inline and simplify result
- Results can change with each debug and optimization flag

It is going to be hard to debug/step through!

Zero Cost - Where's the Code?!

- Inlined functions don't look like the code we wrote
- `const` can allow the compiler to optimize LOCs away
- Static polymorphism and devirtualization inline and simplify result
- Results can change with each debug and optimization flag

It is going to be hard to debug/step through!

count_if

```
template<typename _InputIterator, typename _Predicate>
  typename iterator_traits<_InputIterator>::difference_type
  count_if(_InputIterator __first, _InputIterator __last,
           _Predicate __pred)
  {

    // concept requirements
    __glibcxx_function_requires(_InputIteratorConcept<_InputIterator>)
    __glibcxx_function_requires(_UnaryPredicateConcept<_Predicate,
          typename iterator_traits<_InputIterator>::value_type>)
    __glibcxx_requires_valid_range(__first, __last);
    typename iterator_traits<_InputIterator>::difference_type __n = 0;

    for ( ; __first != __last; ++__first)
      if (__pred(*__first))
        ++__n;
    return __n;

  }
```

count_if

```
template<typename _InputIterator, typename _Predicate>
  typename iterator_traits<_InputIterator>::difference_type
  count_if(_InputIterator __first, _InputIterator __last,
           _Predicate __pred)
  {

    // concept requirements
    __glibcxx_function_requires(_InputIteratorConcept<_InputIterator>)
    __glibcxx_function_requires(_UnaryPredicateConcept<_Predicate,
          typename iterator_traits<_InputIterator>::value_type>)
    __glibcxx_requires_valid_range(__first, __last);
    typename iterator_traits<_InputIterator>::difference_type __n = 0;

    for ( ; __first != __last; ++__first)
      if (__pred(*__first))
        ++__n;
    return __n;

  }
```

count_if

```
template<typename _InputIterator, typename _Predicate>
  typename iterator_traits<_InputIterator>::difference_type
  count_if(_InputIterator __first, _InputIterator __last,
           _Predicate __pred)
  {

    // concept requirements
    __glibcxx_function_requires(_InputIteratorConcept<_InputIterator>)
    __glibcxx_function_requires(_UnaryPredicateConcept<_Predicate,
          typename iterator_traits<_InputIterator>::value_type>)
    __glibcxx_requires_valid_range(__first, __last);
    typename iterator_traits<_InputIterator>::difference_type __n = 0;

    for ( ; __first != __last; ++__first)
      if (__pred(*__first))
        ++__n;
    return __n;

  }
```

count_if

Without optimizations:

```
#include <algorithm>

int main(int argc, char ** argv)
{
    return
      std::count_if( argv+1, argv+argc,
                     [](auto arg)
                     {
                       return std::atoi(arg) == 8;
                     }
                 );
}
```

count_if

```
main:
        push    rbp
        mov     rbp, rsp
        sub     rsp, 32
        mov     DWORD PTR [rbp-20], edi
        mov     QWORD PTR [rbp-32], rsi
        mov     eax, DWORD PTR [rbp-20]
        cdqe
        lea     rdx, [0+rax*8]
        mov     rax, QWORD PTR [rbp-32]
        add     rdx, rax
        mov     rax, QWORD PTR [rbp-32]
        add     rax, 8
        sub     rsp, 8
        push    rcx
        mov     rsi, rdx
        mov     rdi, rax
        call    _ZSt8count_ifIPPcZ4mainEUlT_E_ENSt15iterator_traitsIS2_E15difference_typeES2_S2_T0_
        add     rsp, 16
        leave
        ret
_ZSt8count_ifIPPcZ4mainEUlT_E_ENSt15iterator_traitsIS2_E15difference_typeES2_S2_T0_:
        push    rbp
        mov     rbp, rsp
        push    rbx
        sub     rsp, 24
        mov     QWORD PTR [rbp-24], rdi
        mov     QWORD PTR [rbp-32], rsi
```

count_if

```
        sub     rsp, 8
        push    rax
        call    _ZN9__gnu_cxx5__ops11__pred_iterIZ4mainEUlT_E_EENS0_10_Iter_predIS2_EES2_
        add     rsp, 16
        mov     rdx, QWORD PTR [rbp-32]
        mov     rax, QWORD PTR [rbp-24]
        sub     rsp, 8
        push    rbx
        mov     rsi, rdx
        mov     rdi, rax
        call    _ZSt10__count_ifIPPcN9__gnu_cxx5__ops10_Iter_predIZ4mainEUlT_E_EEENSt15iterator_traitsIS5_E15difference
        add     rsp, 16
        mov     rbx, QWORD PTR [rbp-8]
        leave
        ret
_ZN9__gnu_cxx5__ops11__pred_iterIZ4mainEUlT_E_EENS0_10_Iter_predIS2_EES2_:
        push    rbp
        mov     rbp, rsp
        push    rbx
        sub     rsp, 24
        lea     rax, [rbp-17]
```

count_if

```
        sub     rsp, 8
        push    rdx
        mov     rdi, rax
        call    _ZN9__gnu_cxx5__ops10_Iter_predIZ4mainEUlT_E_EC1ES3_
        add     rsp, 16
        mov     eax, ebx
        mov     rbx, QWORD PTR [rbp-8]
        leave
        ret
_ZSt10__count_ifIPPcN9__gnu_cxx5__ops10_Iter_predIZ4mainEUlT_E_EEENSt15iterator_traitsIS5_E15difference_typeES5_S5_T0_:
        push    rbp
        mov     rbp, rsp
        sub     rsp, 32
        mov     QWORD PTR [rbp-24], rdi
        mov     QWORD PTR [rbp-32], rsi
        mov     QWORD PTR [rbp-8], 0
.L10:
        mov     rax, QWORD PTR [rbp-24]
        cmp     rax, QWORD PTR [rbp-32]
        je      .L8
```

count_if

```
        mov     rax, QWORD PTR [rbp-24]
        mov     rsi, rax
        lea     rdi, [rbp+16]
        call    _ZN9__gnu_cxx5__ops10_Iter_predIZ4mainEUlT_E_EclIPPcEEbS2_
        test    al, al
        je      .L9
        add     QWORD PTR [rbp-8], 1
.L9:
        add     QWORD PTR [rbp-24], 8
        jmp     .L10
.L8:
        mov     rax, QWORD PTR [rbp-8]
        leave
        ret
_ZN9__gnu_cxx5__ops10_Iter_predIZ4mainEUlT_E_EC2ES3_:
        push    rbp
        mov     rbp, rsp
        mov     QWORD PTR [rbp-8], rdi
        nop
        pop     rbp
        ret
_ZZ4mainENKUlT_E_clIPcEEDaS_:
        push    rbp
        mov     rbp, rsp
        sub     rsp, 16
        mov     QWORD PTR [rbp-8], rdi
        mov     QWORD PTR [rbp-16], rsi
```

count_if

```
        mov     rax, QWORD PTR [rbp-16]
        mov     rdi, rax
        call    atoi
        cmp     eax, 8
        sete    al
        leave
        ret
_ZN9__gnu_cxx5__ops10_Iter_predIZ4mainEUlT_E_ElIPPcEEbS2_:
        push    rbp
        mov     rbp, rsp
        sub     rsp, 16
        mov     QWORD PTR [rbp-8], rdi
        mov     QWORD PTR [rbp-16], rsi
        mov     rax, QWORD PTR [rbp-16]
        mov     rdx, QWORD PTR [rax]
        mov     rax, QWORD PTR [rbp-8]
        mov     rsi, rdx
        mov     rdi, rax
        call    _ZZ4mainENKUlT_E_clIPcEEDaS_
        leave
        ret
```

Zero Cost - Where's the Code?!

Challenges writing and debugging.

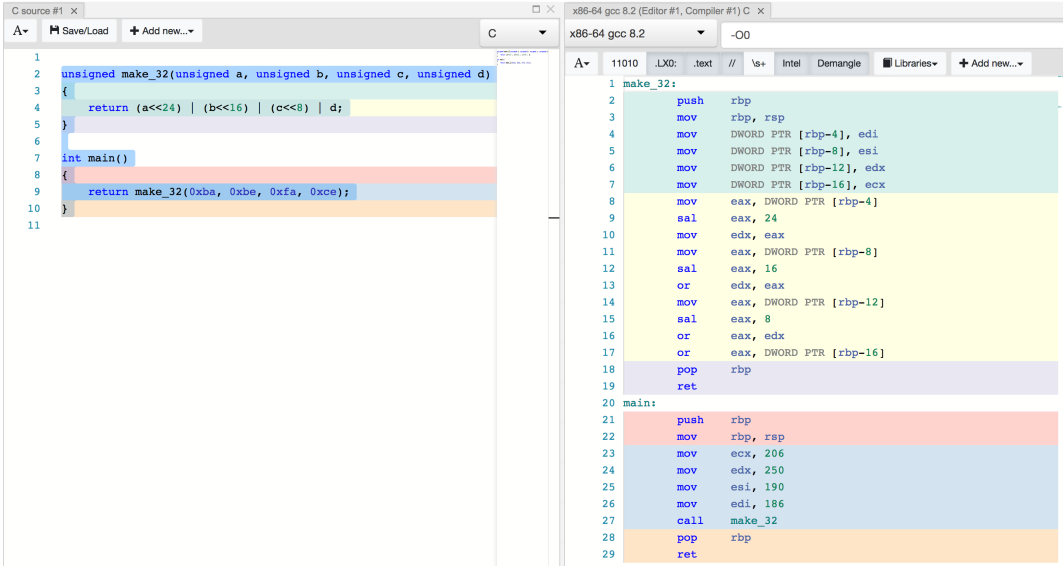
Optimization Flags

```
-O0
Reduce compilation time and make debugging
produce the expected results. This is the default.
```

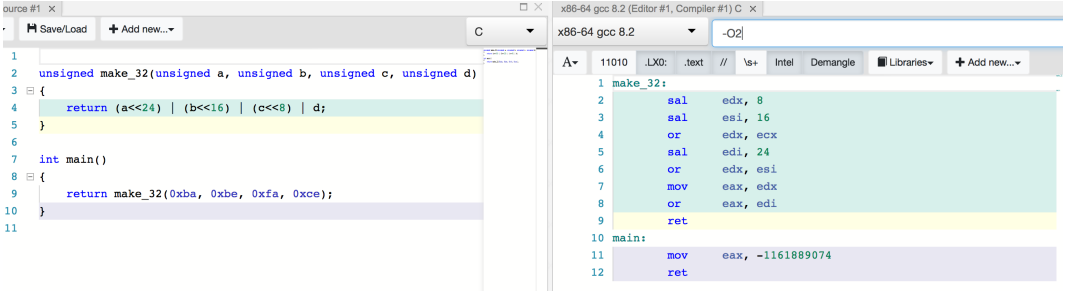
C and Optimizations

```
unsigned make_32(unsigned a, unsigned b, unsigned c, unsigned d)
{
    return (a<<24) | (b<<16) | (c<<8) | d;
}

int main()
{
    return make_32(0xba, 0xbe, 0xfa, 0xce);
}
```

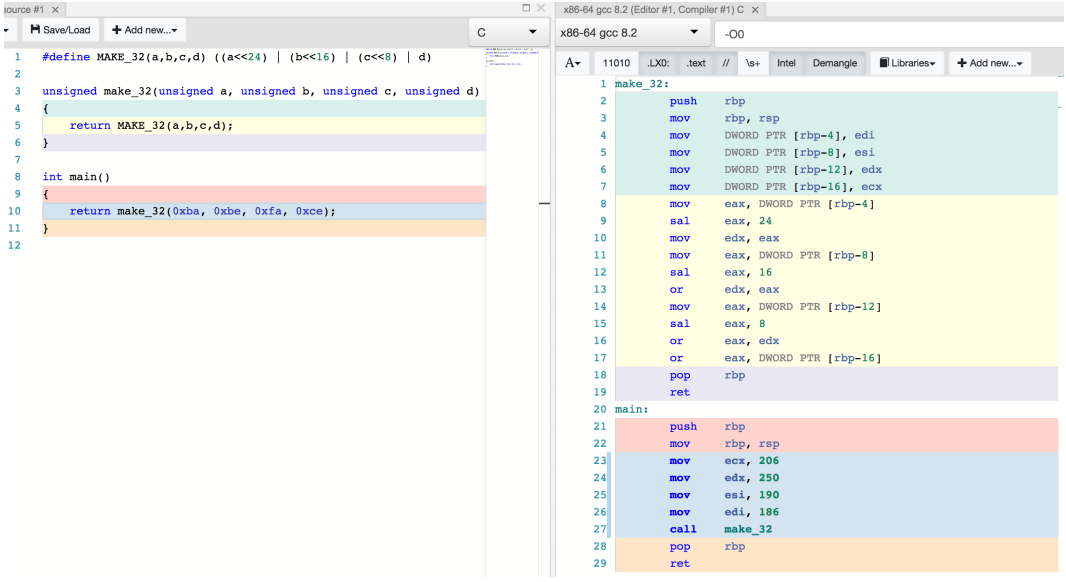
C and Optimizations



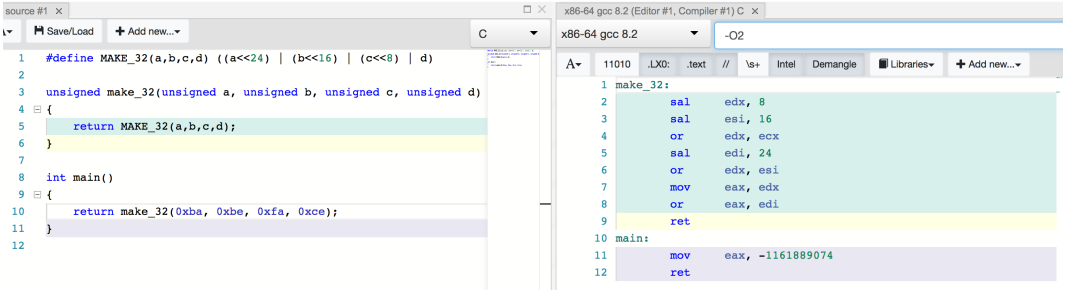
C and Optimizations



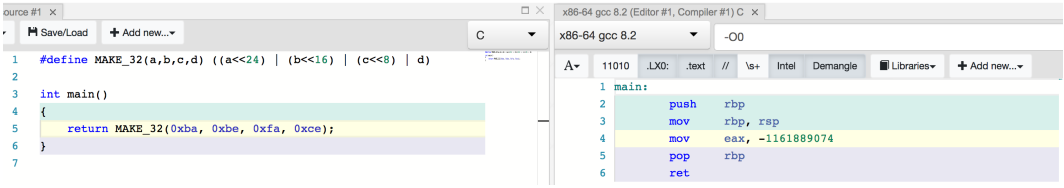
C and Optimizations



C and Optimizations



C and Optimizations



C and Optimizations

`-1161889074` is `0xffffffffbabeface`

Optimizations

- `-O0` might be even more pessimistic than you think
- Don't pretend to know what the compiler will do.

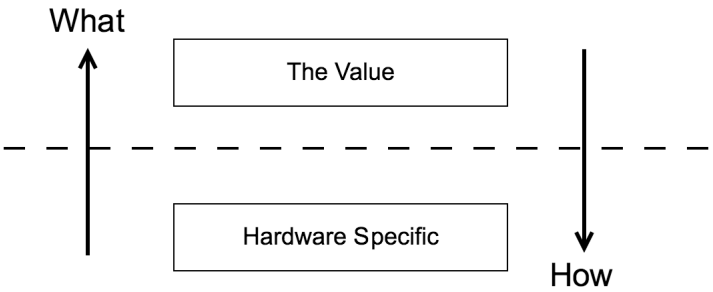
Zero Cost - Where's the Code?!

Challenges writing and debugging.

REPL

Read-**E**val-**P**rint **L**oop

Host Debugging



Optimization Flags

```
-O0
Reduce compilation time and make debugging
produce the expected results. This is the default.
```

`goto` error handling idiom

```
int important_function(int foo)
{
    int return_value = 0;
    if (!init_stuff(foo)) {
        goto error_1;
    }
    if (!prepare_stuff(foo)) {
        goto error_2;
    }
    if (do_stuff(foo))
    {
        return_value = get_result(foo);
    }

    cleanup_3();
error_2:
    cleanup_2();
error_1:
    cleanup_1();
    return return_value;
}
```

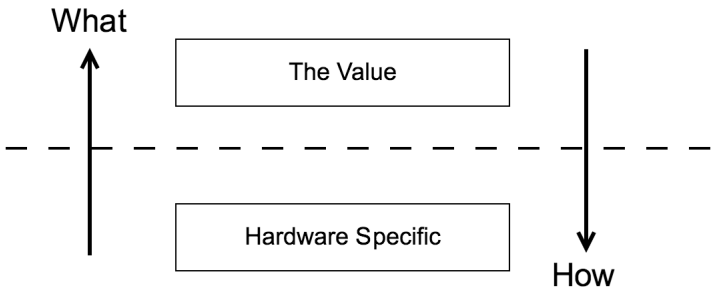
C++ is not C

- C++ is not C
- Languages have idioms
- Think higher without sacrifice

Correctness

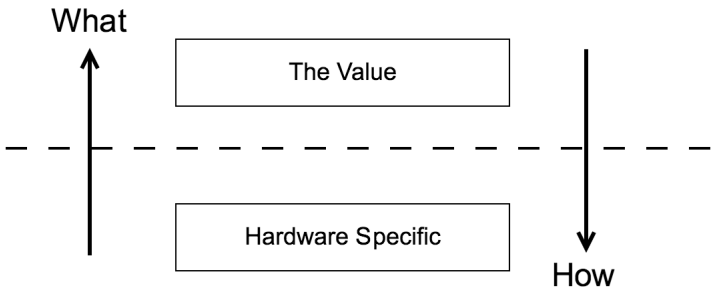
- Type system enforced correctness
- Move as much checking to compile time

Absractions



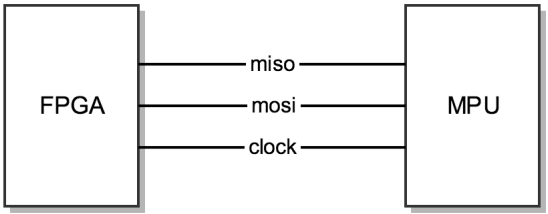
- The good
- The bad

Absractions



- The good
- ~~The bad~~

SPI Serialization



Messaging in C

```
int setThing(struct mcu_s *mcu, int thing, uint16_t value, callback cb, void *ctx)
{
   unsigned length = 4;
   uint8_t data[length];

   data[0] = SET_THING_CMD;
   data[1] = thing;
   memcpy(&data[2], &value, 2);

   return request(mcu->spi_link, data, length, cb, ctx);
}


int setThingMode(struct mcu_s *mcu, int thing, int mode, callback cb, void *ctx)
{
   unsigned length = 3;
   uint8_t data[length];

   data[0] = SET_THING_MODE_CMD;
   data[1] = thing;
   data[2] = mode;

   return request(mcu->spi_link, data, length, cb, ctx);
}
```

Messaging in C++

```cpp
// protocol.hpp

namespace protocol
{
   enum class thing_id : uint8_t
   {
      left, right, other
   };

   struct set_thing
   {
      thing_id thing;
      uint16_t value;
   };

   struct set_thing_mode
   {
      thing_id thing;
      bool mode;
   };
}
```

```cpp
// protocol_adapted.hpp

CIERELABS_ADAPT_STRUCT(
   protocol::set_thing,
   (thing)(value) )

CIERELABS_ADAPT_STRUCT(
   protocol::set_thing_mode,
   (thing)(mode) )

namespace protocol
{
   using message_list = meta_list<
      protocol::set_thing,
      protocol::set_thing_mode
      >;
}
```

Messaging in C++

```cpp
// protocol.hpp

namespace protocol
{
   enum class thing_id : uint8_t
   {
      left, right, other
   };

   struct set_thing
   {
      thing_id thing;
      uint16_t value;
   };

   struct set_thing_mode
   {
      thing_id thing;
      bool mode;
   };
}
```

```cpp
// protocol_adapted.hpp

CIERELABS_ADAPT_STRUCT(
   protocol::set_thing,
   (thing)(value) )

CIERELABS_ADAPT_STRUCT(
   protocol::set_thing_mode,
   (thing)(mode) )

namespace protocol
{
   using message_list = meta_list<
      protocol::set_thing,
      protocol::set_thing_mode
      >;
}
```

Messaging in C++

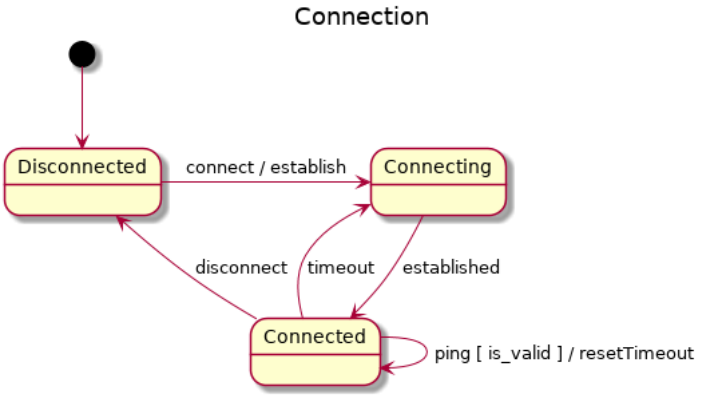
```
protocol::set_thing_mode set_thing_mode{ protocol::thing_id::other,
                                         false };

// ...

send_message(set_thing_mode);
```

Abstraction - SPI Serialization

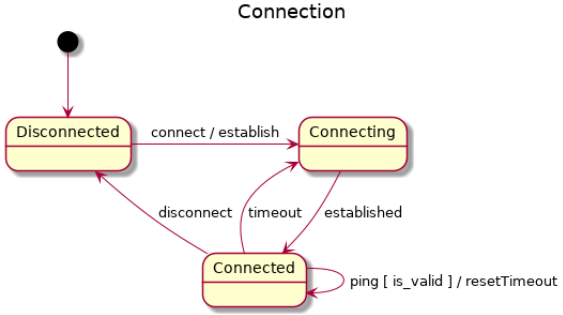
C Version LOC : 3292

C++ Version LOC : 1194

Absractions - Boost.SML



Absractions - Boost.SML




```
sml::sm connection_machine = [] {
   return transition_table{
      * "Disconnected"_s  + connect / establish                = "Connecting"_s    ,
        "Connecting"_s    + established                        = "Connected"_s     ,
        "Connected"_s     + ping [is_valid]  / resetTimeout                        ,
        "Connected"_s     + timeout / establish                = "Connecting"_s    ,
        "Connected"_s     + disconnect / close                 = "Disconnected"_s
   };
};
```

Abstractions - Boost.SML

```
sml::sm connection_machine = [] {
   return transition_table{
      * "Disconnected"_s  + connect / establish              = "Connecting"_s   ,
        "Connecting"_s    + established                      = "Connected"_s    ,
        "Connected"_s     + ping [is_valid]  / resetTimeout                     ,
        "Connected"_s     + timeout / establish              = "Connecting"_s   ,
        "Connected"_s     + disconnect / close               = "Disconnected"_s
   };
};


const auto establish  = []{ /* something */ };
const auto disconnect = []{ /* something */ };


const auto is_valid   = [](auto event){ return true; };
```

Abstractions - Boost.SML

```
sml::sm connection_machine = [] {
   return transition_table{
      * "Disconnected"_s  + connect / establish              = "Connecting"_s   ,
        "Connecting"_s    + established                      = "Connected"_s    ,
        "Connected"_s     + ping [is_valid]  / resetTimeout                     ,
        "Connected"_s     + timeout / establish              = "Connecting"_s   ,
        "Connected"_s     + disconnect / close               = "Disconnected"_s
   };
};
```

```
const auto establish  = []{ /* something */ };
const auto disconnect = []{ /* something */ };
```

```
const auto is_valid   = [](auto event){ return true; };
```

Abstractions - Boost.SML

```
sml::sm connection_machine = [] {
   return transition_table{
      * "Disconnected"_s  + connect / establish              = "Connecting"_s   ,
        "Connecting"_s    + established                      = "Connected"_s    ,
        "Connected"_s     + ping [is_valid]  / resetTimeout                     ,
        "Connected"_s     + timeout / establish              = "Connecting"_s   ,
        "Connected"_s     + disconnect / close               = "Disconnected"_s
   };
};
```

```
const auto establish  = []{ /* something */ };
const auto disconnect = []{ /* something */ };
```

```
const auto is_valid   = [](auto event){ return true; };
```

Abstractions - Boost.SML

Usage:

```
connection_machine.process_event(connect{});
```

Absractions - Boost.SML

Technique	SM Size (bytes)
Naive	3
Enum/Switch	1
SML	1

Kris Jusiak

https://github.com/boost-experimental/sml

Absractions - Boost.SML

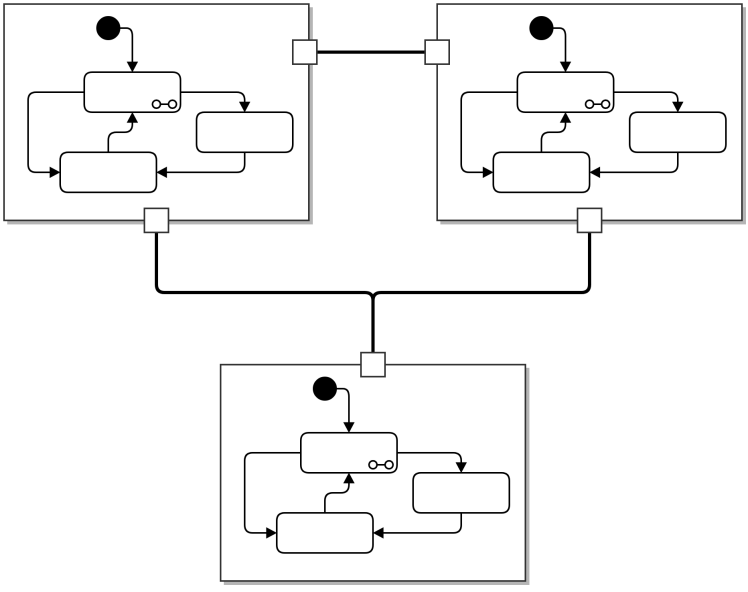
Technique	SM Size (bytes)
Naive	3
Enum/Switch	1
SML	1

Kris Jusiak

`https://github.com/boost-experimental/sml`

ciere.com

Absractions - Ladon



Abstractions - Ladon

```
namespace stepper_msg
{
   struct home              {};
   struct set_position      { size_t location; };
   struct move_absolute     { size_t location; };
   struct move_relative     { int steps; };
   struct drive             { bool forward; };
   struct halt              {};
   struct set_velocity      { unsigned int vel; };
   struct set_acceleration  { unsigned int acc; };
   struct get_position      {};

   struct move_done         {};
   struct velocity_set      {};
   struct move_error        { std::string error; };
   struct halted            {};
   struct comm_error        { std::string error; };
   struct cmd_done          { bool pass; };
   struct position          { size_t location; };
}
```

Abstractions - Ladon

```
struct stepper_protocol_t
   : ciere::ladon::protocol< meta_list<  // in signals
                                stepper_msg::home
                              , stepper_msg::set_position
                              , stepper_msg::move_absolute
                              , stepper_msg::move_relative
                              , stepper_msg::drive
                              , stepper_msg::halt
                              , stepper_msg::set_velocity
                              , stepper_msg::set_acceleration
                              , stepper_msg::get_position >
                           , meta_list< // out signals
                                stepper_msg::move_done
                              , stepper_msg::move_error
                              , stepper_msg::comm_error
                              , stepper_msg::cmd_done
                              , stepper_msg::halted
                              , stepper_msg::velocity_set
                              , stepper_msg::position > >
{};
```

Abstractions - Ladon

```
// ------------------------------------------
// tags for ports
// ------------------------------------------
struct app_port         {};
struct stepper_port     {};
struct timer_port       {};
// ------------------------------------------
// ------------------------------------------

// ------------------------------------------
// list of ports
// ------------------------------------------
//
using port_list =
   meta_list<
//       Port             Protocol
//     +--------------+----------
   Port< app_port      , stepper_protocol_t         > ,
   Port< stepper_port  , stepper_motor_protocol_t   > ,
   Port< timer_port    , timer_protocol_t           >
//       +-------------+----------
   >;
// ------------------------------------------
```

Absractions - Ladon

- Bind ports at compile time
- Bind ports at run time
- Compile time checking
- Policies at “Controller”
- Bind ports to hardware!

Absractions - Odin Holmes' work

9.2.2 Module Stop Control Register B (MSTPCRB)

The MSTPCRB register controls the module-stop state.
For release from the module-stop state, see section 9.3.1, Module-Stop Function.

Address(es): A00B 0304h

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	MSTPCRB19	MSTPCRB18*1	MSTPCRB17	MSTPCRB16
Value after reset:	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MSTPCRB15*1	MSTPCRB14	MSTPCRB13	MSTPCRB12	MSTPCRB11	MSTPCRB10	MSTPCRB9	MSTPCRB8	MSTPCRB7	MSTPCRB6	MSTPCRB5	—	MSTPCRB3	MSTPCRB2	MSTPCRB1	—
Value after reset:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Note 1. These bits are only for products incorporating an EtherCAT (optional).

Bit	Symbol	Bit Name	Description	R/W
b0	—	Reserved	This bit is read as 0. The write value should be 0.	R/W
b1	MSTPCRB1	RSCAN Module Stop	Target module: RSCAN 0: Release from the module-stop state 1: Transition to the module-stop state is made	R/W
b2	MSTPCRB2	RIICa Unit 1 Module Stop	Target module: RIICa unit 1 0: Release from the module-stop state 1: Transition to the module-stop state is made	R/W
b3	MSTPCRB3	RIICa Unit 0 Module Stop	Target module: RIICa unit 0 0: Release from the module-stop state 1: Transition to the module-stop state is made	R/W
b4	—	Reserved	This bit is read as 1.	R/W

- Watch C++Now 2018 video on Mixins
- FSRs

C++ Features

- Lambda Expressions
- Algorithms
- Range-based for loops
- Move semantics
- fold expressions
- `initializer_list`

Part IV

Thoughts

Thoughts

- Tool vendors aren't friendly to C++
- Abstractions are wonderful!
- Use good tools ... including Compiler Explorer
- Measure
- Test

Questions?

Questions?