Clang on a SoC

Using Clang on a BeagleBone Black to run a cape

Cheinan Marks

std::disclaimer<! I.hardware_guy()>;

SoC = System on a Chip

Hardware: https://upverter.com/cheinan

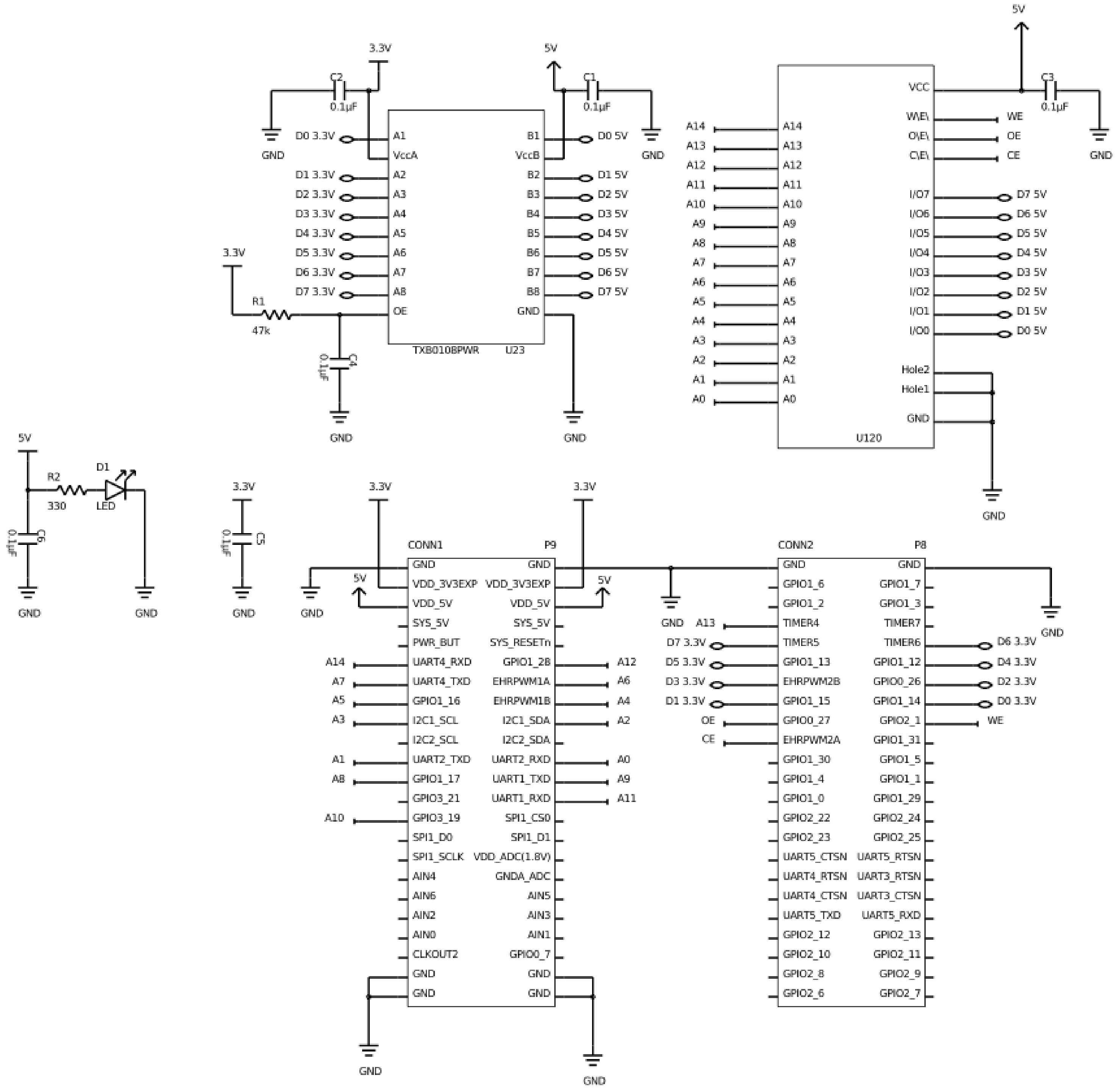
Software: https://github.com/cheinan/eeprom

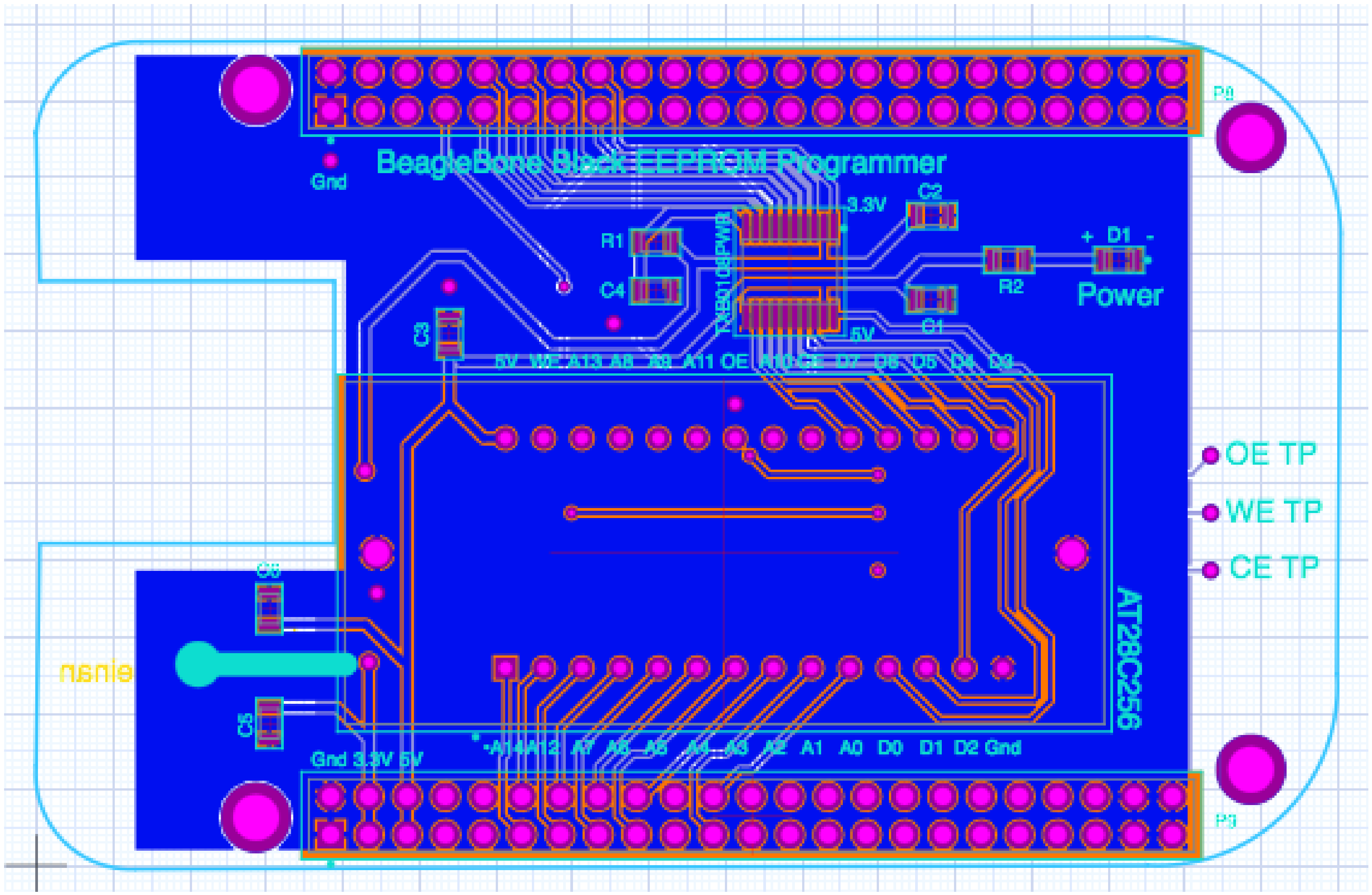
A Long Long Time Ago

http://pixabay.com/en/party-dancing-dancer-disco-146582/ CC0 Public Domain

The Mission

- Build a Single Board Z-80 Computer
- Use an EEPROM to hold software
- Assemble software on modern computer
- Burn machine code to EEPROM
- Need an EEPROM burner
- Spend $150 or build a burner from scratch?

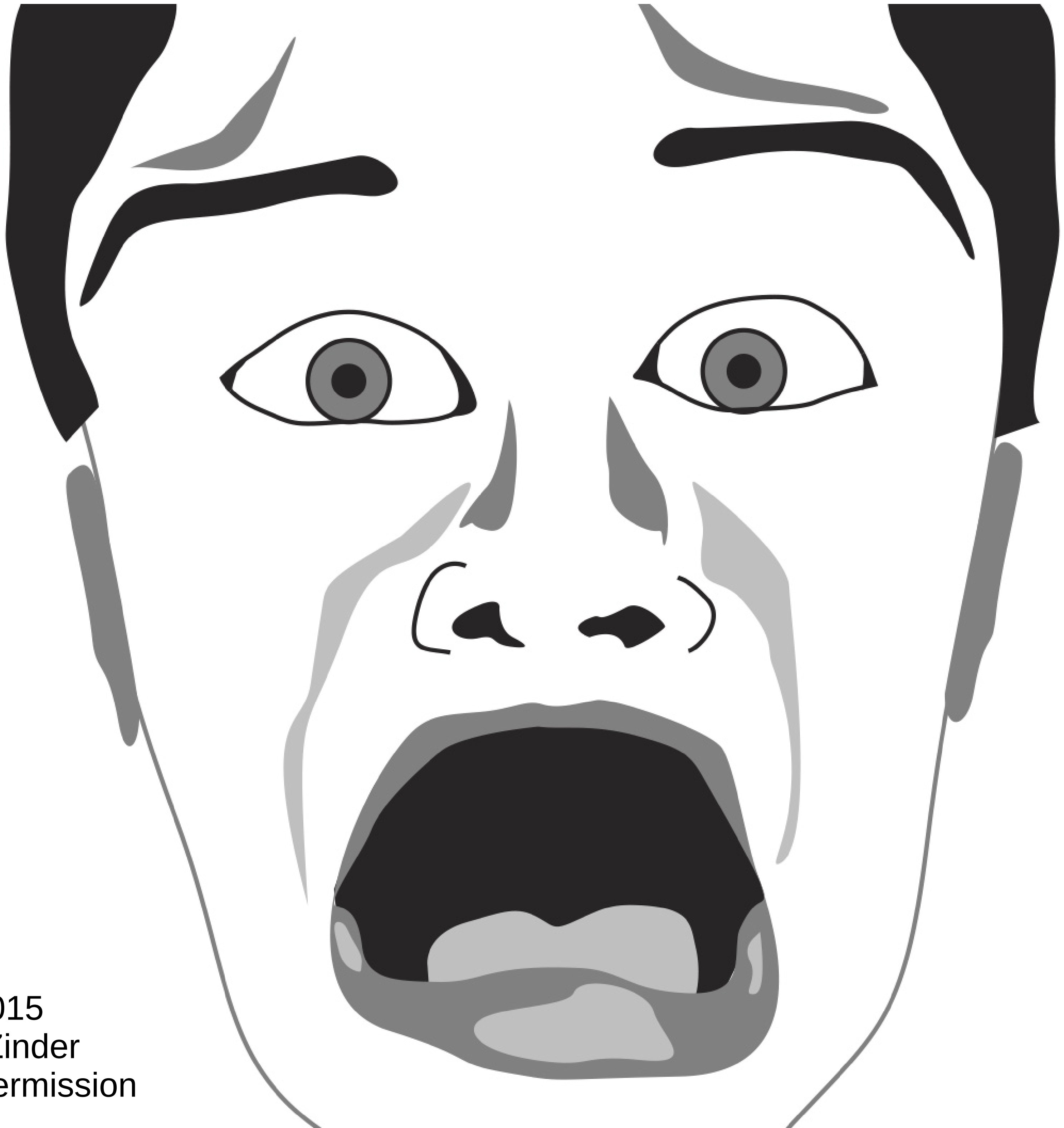

3.3V
5V
GND
C1
C2
C3
C4
C5
C6
0.1µF
R1
47k
R2
330
D1
LED
TXB0108PWR
U23
A1
VccA
A2
A3
A4
A5
A6
A7
A8
OE
B1
VccB
B2
B3
B4
B5
B6
B7
B8
D0 3.3V
D1 3.3V
D2 3.3V
D3 3.3V
D4 3.3V
D5 3.3V
D6 3.3V
D7 3.3V
D0 5V
D1 5V
D2 5V
D3 5V
D4 5V
D5 5V
D6 5V
D7 5V
U120
VCC
W\E\
O\E\
C\E\
WE
OE
CE
I/O7
I/O6
I/O5
I/O4
I/O3
I/O2
I/O1
I/O0
Hole2
Hole1
A14
A13
A12
A11
A10
A9
A0
CONN1
P9
CONN2
P8
VDD_3V3EXP
VDD_5V
SYS_5V
PWR_BUT
SYS_RESETn
UART4_RXD
UART4_TXD
GPIO1_28
EHRPWM1A
GPIO1_16
EHRPWM1B
I2C1_SCL
I2C1_SDA
I2C2_SCL
I2C2_SDA
UART2_TXD
UART2_RXD
GPIO1_17
UART1_TXD
GPIO3_21
UART1_RXD
GPIO3_19
SPI1_CS0
SPI1_D0
SPI1_D1
SPI1_SCLK
VDD_ADC(1.8V)
AIN4
GNDA_ADC
AIN6
AIN5
AIN2
AIN3
AIN0
AIN1
CLKOUT2
GPIO0_7
GPIO1_6
GPIO1_7
GPIO1_2
GPIO1_3
TIMER4
TIMER7
TIMER5
TIMER6
GPIO1_13
GPIO1_12
EHRPWM2B
GPIO0_26
GPIO1_15
GPIO1_14
GPIO0_27
GPIO2_1
EHRPWM2A
GPIO1_31
GPIO1_30
GPIO1_5
GPIO1_4
GPIO1_1
GPIO1_0
GPIO1_29
GPIO2_22
GPIO2_24
GPIO2_23
GPIO2_25
UART5_CTSN
UART5_RTSN
UART4_RTSN
UART3_RTSN
UART4_CTSN
UART3_CTSN
UART5_TXD
UART5_RXD
GPIO2_12
GPIO2_13
GPIO2_10
GPIO2_11
GPIO2_8
GPIO2_9
GPIO2_6
GPIO2_7


BeagleBone Black EEPROM Programmer
P9
P8
Gnd
3.3V
C2
R1
TXB0108PWR
+ D1 -
R2
Power
C4
C1
C3
5V
5V WE A13 A8 A9 A11 OE A10 CE D7 D6 D5 D4 D3
OE TP
WE TP
CE TP
AT28C256
C6
C5
Gnd 3.3V 5V
A14 A12 A7 A6 A5 A4 A3 A2 A1 A0 D0 D1 D2 Gnd

```
int gpio_unexport(unsigned int gpio)
{
    int fd, len;
    char buf[MAX_BUF];

    fd = open(SYSFS_GPIO_DIR "/unexport",
O_WRONLY);
    if (fd < 0) {
        perror("gpio/export");
        return fd;
    }

    len = snprintf(buf, sizeof(buf), "%d", gpio);
    write(fd, buf, len);
    close(fd);
    return 0;
}
```



I Want my C++!

- Download LLVM/Clang
- Configuration autodetects ARM architecture
- Build
- Install
- Write Modern C++

RAII

```
DataBus::DataBus(bool is_data_out) : m_is_data_out(is_data_out)
{
    for (const auto gpio : m_data_gpio) {
    gpio_export(gpio);
    gpio_set_dir(gpio, is_data_out ? OUTPUT_PIN : INPUT_PIN);
  }
}
DataBus::~DataBus()
{
  for (const auto gpio : m_data_gpio) {
    gpio_unexport(gpio);
  }
}
```

STL & Move Semantics

```
std::vector<unsigned char> ReadBlock(unsigned short
address, unsigned short length);
```

Exceptions

```
if (! m_is_data_out) {
  throw EEPROMException("Tried to write to data bus
that is configured for reading");
}
```

Plus much more C++11 goodness

Go Forth & Make!*

*The above does not refer to two computer languages and a build program.