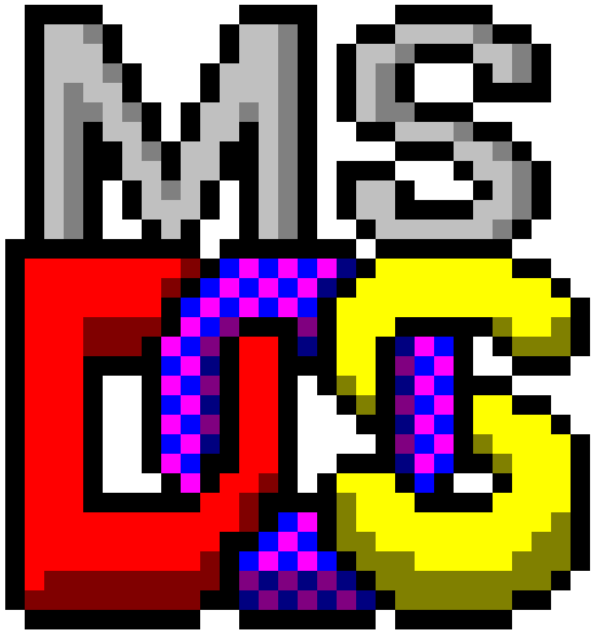
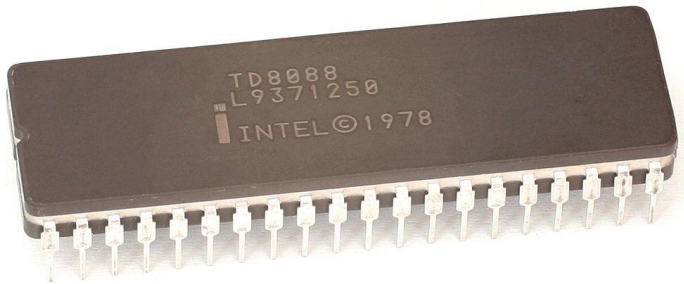
16-bit C++

Considerations for new code on ancient hardware

Jake S. Del Mastro





WARNING: PLEASE DO NOT USE THIS AS GENERAL C++ ADVICE

What do we want?	But how do we do that?
• Good performance (speed) • Not to run out of memory • Well defined behaviour	• Keep code small (48k) • Avoid accessing memory • Use your instruction set • Read the output • Simple solutions are usually the best

Make sure you know what you're doing

```
//int == 16 bits
const int screenHeight = 200;
const int screenWidth = 320;

if (x < 0 || y < 0) return;

int pixelIndex = y * screenWidth + x;

//better, no overflow
const int16_t screenHeight = 200;
const int16_t screenWidth = 320;

if (x < 0 || y < 0) return;

uint16_t pixelIndex = (uint16_t)y * screenWidth + x;
```

Design the code around the machine

```
//fill function table for OPL driver
static void(near* const near opl_funcs[8]) (uint8_t, uint8_t, uint8_t) = { OPL_note_off,
OPL_note_on, nothing, OPL_handle_cc, OPL_program_change, nothing, nothing, nothing };

_FAR_SS void handle_midi_Message() {
        do {
                //Read message & stuff...

                const uint8_t arg0 = *(midiBuf++);
                const uint8_t arg1 = *midiBuf;

                //execute the action from the function table
                midi_funcs[message & 0x07](status, arg0, arg1);

                //Do a bit more stuff...

                midi_message_timer = read_var_len(midiBuf); //set timer for next event
        } while (midi_message_timer == 0);

        midi_sound::current_trackPtr = midiBuf; //update the index
}
```

Break the type system when it helps

```
//just two lists of pointers, but the template will be instantiated twice
std::vector<gameObject*>         gameObject::object_table;
std::vector<scriptProcess*>      scriptProcess::script_table;

//only instantiated once
std::vector<void*>               gameObject::object_table;
std::vector<void*>               scriptProcess::script_table;

inline scriptProcess& getProcess(scriptProcess::processID s) {
        return *static_cast<scriptProcess*>(scriptProcess::script_table[s]);
}

inline gameObject& getObjectByID(gameObject::idType o) {
        return *static_cast<gameObject*>(gameObject::object_table[o]);
}
```

Fancy CPU Instructions

- `rep movsb` is probably the greatest CPU instruction ever invented
- How can we use it?
- memcpy() and friends

```
//this needs to be fast because it needs to finish before the retrace finishes
void colorPalette::updateRange(uint8_t startIndex, const uint8_t* values, uint16_t length)
{
        uint8_t* pPtr = paletteBuffer + startIndex * 3;

        memcpy(pPtr, values, length); //memcpy to the rescue

        numModified = int_max((int16_t)startIndex * 3 + length, numModified);
}
```

C++ isn't always the answer

- I was super sad when I realized this
- Write pure assembly when necessary (e.g. graphics functions)
- Be creative: I designed an entire language to compress the logic required for adventure games

```
//chuckscript
if selected_object == NONE
{
        use_hands arg_1 this
}
else if selected_object == gas_can
{
        car_fueled = true
        remove_from_inventory arg_1 gas_can
        selected_object = NONE
        play_sound did_something_sound
}
```

Plugs

- Find me online:

Twitter: @jakedelmastro

Web: jakedelmastro.com

Demos: pgram.itch.io

GitHub: github.com/pgrAm

- Special thanks to:

OpenWatcomV2 Project