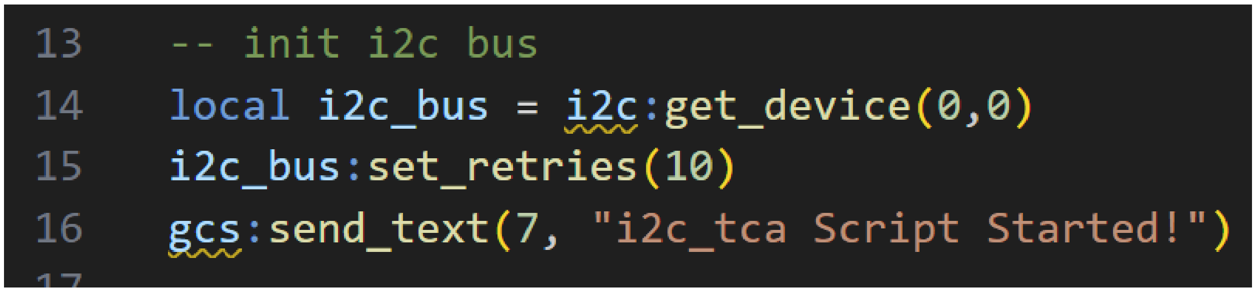
Ryan Prince
UAVLab i2c_tca Documentation

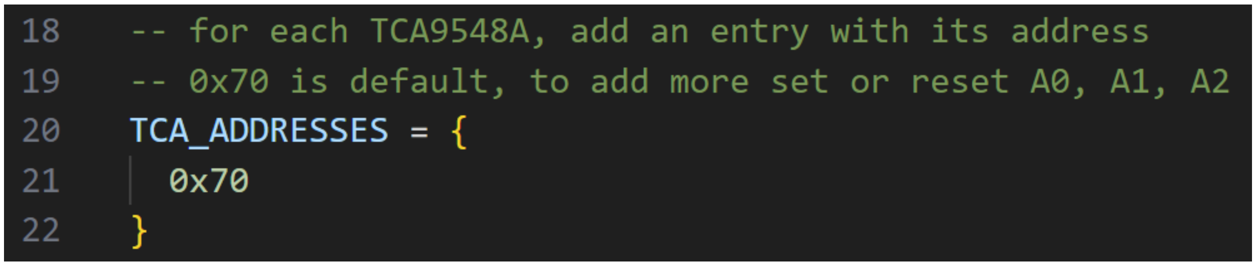
This document intends to explain the code written to use the TCA9548 multiplexer with sensors. The multiplexer is used so that multiple sensors <u>with the same address</u> can be used at the same time.

Starting from the top down:



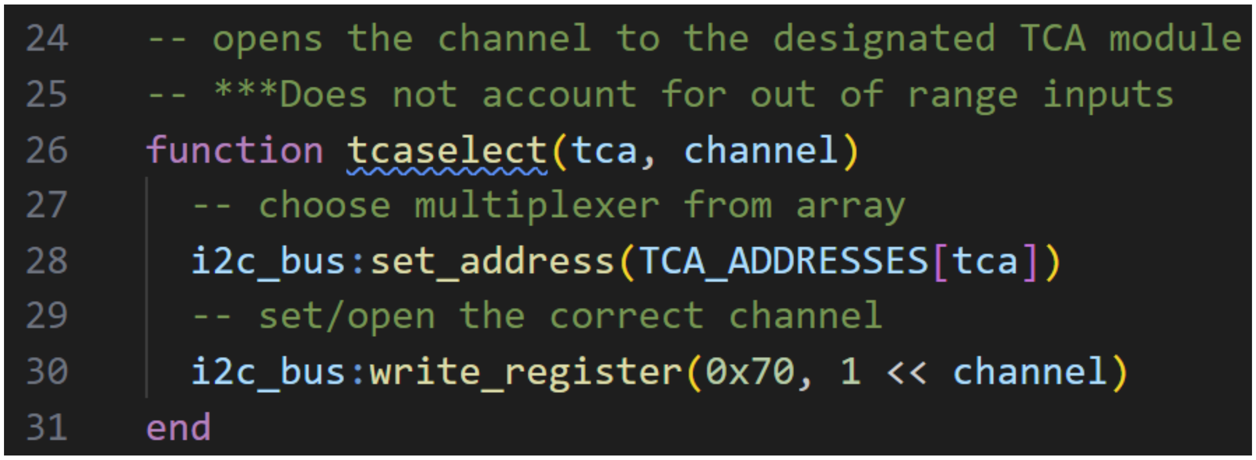
```lua
-- init i2c bus
local i2c_bus = i2c:get_device(0,0)
i2c_bus:set_retries(10)
gcs:send_text(7, "i2c_tca Script Started!")
```

These lines set up variables that will be used throughout the code. “i2c_bus” is the data stream that we will be communicating on to talk to the sensors and the TCA module. Setting retries is just in-case the bus isn’t immediately detected. And then a debug message is sent to the console to let us know the script is running.



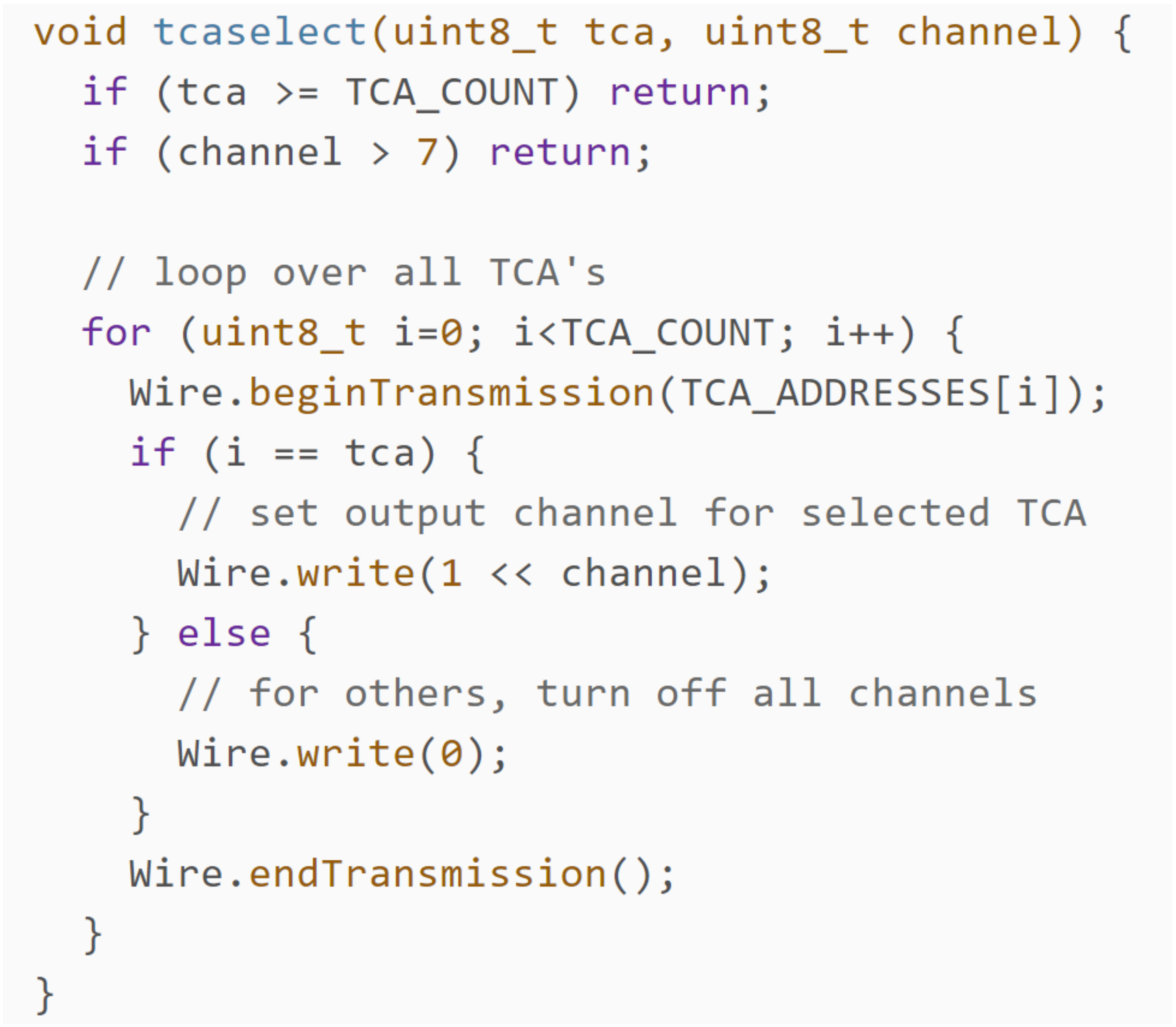
```lua
-- for each TCA9548A, add an entry with its address
-- 0x70 is default, to add more set or reset A0, A1, A2
TCA_ADDRESSES = {
  0x70
}
```

This variable is the array of TCA module addresses. 0x70 is the default and it can be changed or more can be added by setting or resetting A0-A2. There are 8 channels per multiplexer at a max of 8 multipliers per i2c bus.



```lua
-- opens the channel to the designated TCA module
-- ***Does not account for out of range inputs
function tcaselect(tca, channel)
  -- choose multiplexer from array
  i2c_bus:set_address(TCA_ADDRESSES[tca])
  -- set/open the correct channel
  i2c_bus:write_register(0x70, 1 << channel)
end
```

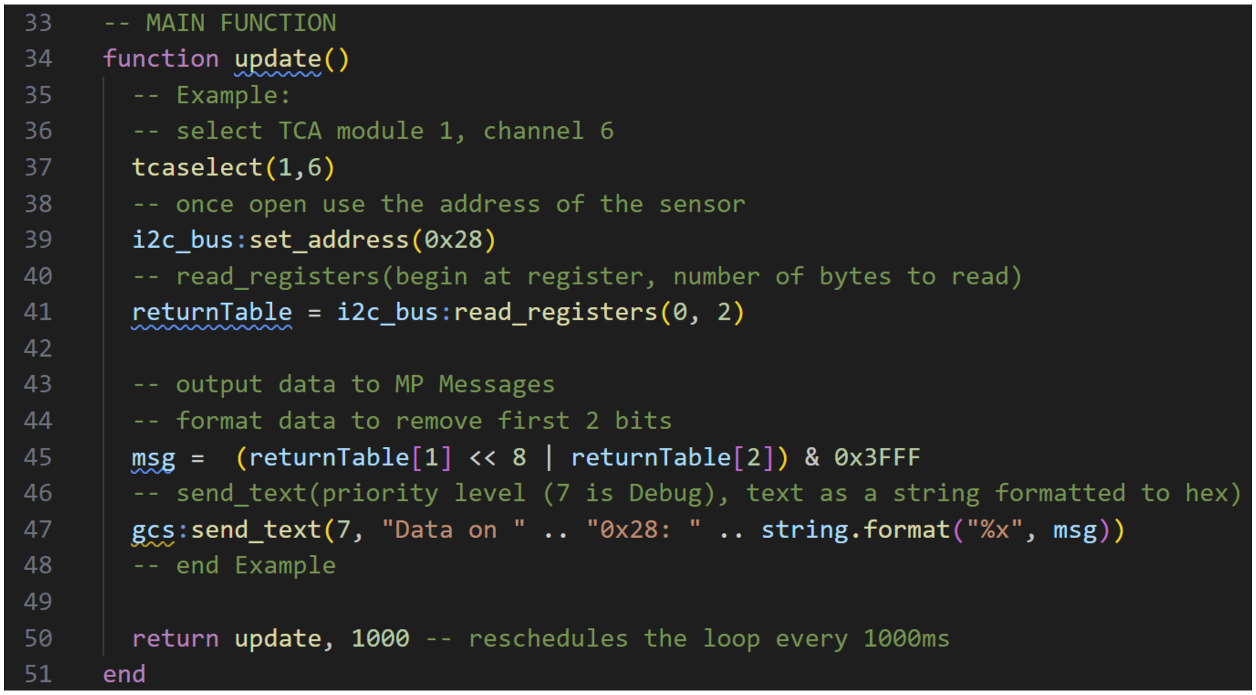
This function selects a channel from the TCA module to communicate with. It accomplishes this by writing a 1 to the corresponding bit, i.e. bit0 opens channel 0, bit1 opens channel 1, etc. This function was adapted from the given arduino function. The lua function does not account for out of range inputs whereas the arduino one does. The arduino function is shown below:



```
void tcaselect(uint8_t tca, uint8_t channel) {
  if (tca >= TCA_COUNT) return;
  if (channel > 7) return;

  // loop over all TCA's
  for (uint8_t i=0; i<TCA_COUNT; i++) {
    Wire.beginTransmission(TCA_ADDRESSES[i]);
    if (i == tca) {
      // set output channel for selected TCA
      Wire.write(1 << channel);
    } else {
      // for others, turn off all channels
      Wire.write(0);
    }
    Wire.endTransmission();
  }
}
```

With this function it is as easy as choosing the channel to communicate with and start reading data.


```lua
-- MAIN FUNCTION
function update()
  -- Example:
  -- select TCA module 1, channel 6
  tcaselect(1,6)
  -- once open use the address of the sensor
  i2c_bus:set_address(0x28)
  -- read_registers(begin at register, number of bytes to read)
  returnTable = i2c_bus:read_registers(0, 2)

  -- output data to MP Messages
  -- format data to remove first 2 bits
  msg =  (returnTable[1] << 8 | returnTable[2]) & 0x3FFF
  -- send_text(priority level (7 is Debug), text as a string formatted to hex)
  gcs:send_text(7, "Data on " .. "0x28: " .. string.format("%x", msg))
  -- end Example

  return update, 1000 -- reschedules the loop every 1000ms
end
```

This example program was written to show how to communicate with one sensor from one TCA module on one channel. This example program assumes 0x28 is the address of the sensor on channel 6 of the first TCA module. This simply just gets data and outputs it to the log. The next step from this would be to write to the .bin file which, how to do that is outlined in the i2c_readSensors documentation.

References
Ardupilot lua function definition:
https://github.com/ArduPilot/ardupilot/blob/master/libraries/AP_Scripting/docs/docs.lua
TCA9548 datasheet: https://cdn-shop.adafruit.com/datasheets/tca9548a.pdf
TCA9548 arduino scripts: https://learn.adafruit.com/working-with-multiple-i2c-devices/arduino-5
TCA9548 store page: https://www.adafruit.com/product/2717
Cube orange documentation:
https://docs.cubepilot.org/user-guides/autopilot/the-cube-module-overview