

# Engineering Portfolio

Haya Dakhil



# Concentray: PCB

Embedded controller / display board for a desk-mounted focus device with screen timers and task tracking.

*Schematic & PCB design, simulations, component selection & BOM, assembly, bring-up, documentation.*

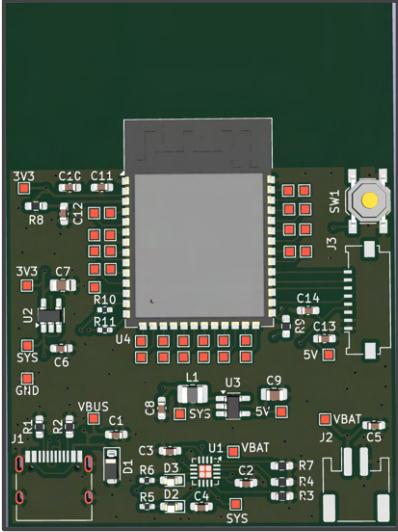
**Design includes:** 4-layer PCB, ESD protection, DC-DC Buck (3.3V) and Boost (5V) Regulators, SPI communication, Smart power path management, header (V1) / FFC (V2) connectors, SWD, via stitching.

**Tools used:** KiCad, LTSpice, EE lab tools (oscilloscope, DMM, power supply, LPKF Reflow, SMT/stencil soldering).

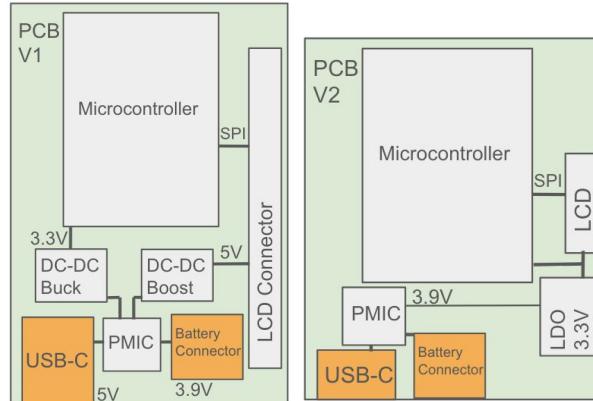
V1



V2



## Circuit Block Diagrams





# Concentray: Firmware

**UI/UX firmware, validated with simulation to bypass the need for microcontroller hardware**

*Wireframing, UI/UX principles, widget drawing, navigation logic, simulation, documentation*

**Firmware includes:** Timer countdown, scrolling navigation, button animations, start/stop/pause/resume.

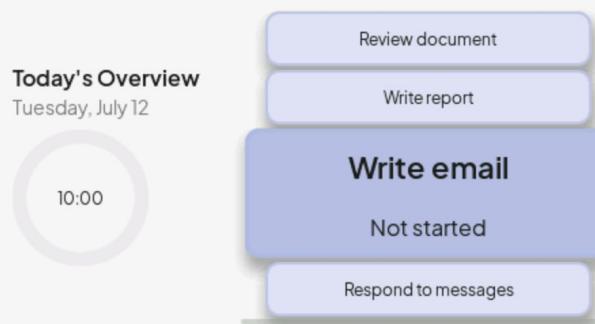
**Software / Frameworks:** C++, LVGL graphics library, Arduino, PlatformIO, SDL.



You have  
4 tasks today

Press button to view tasks

Let's get something done today.



## Review document

Review the latest draft and leave concise comments.



Blocked

### Pause Task

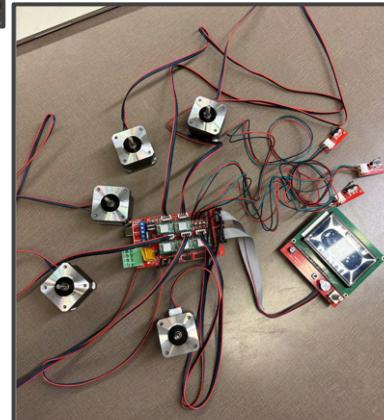
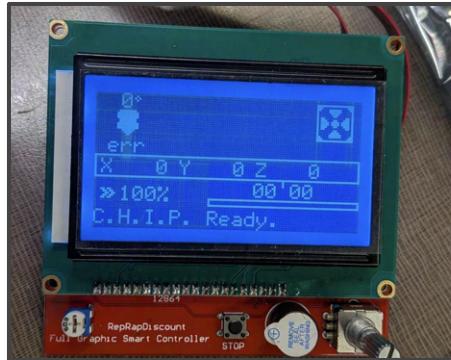
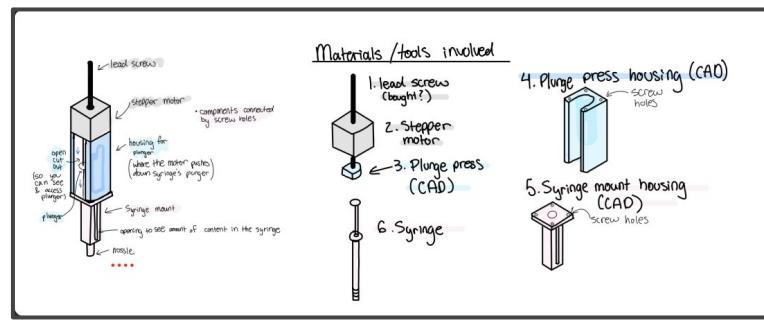
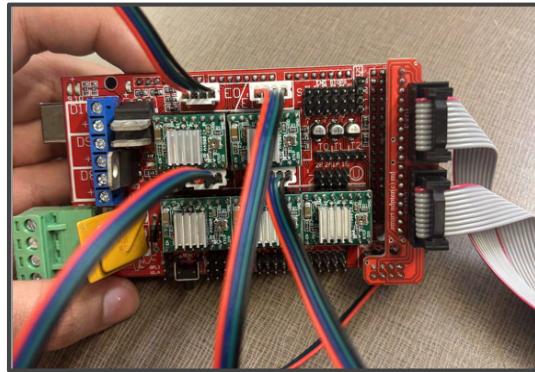
```
 0 @ main.cpp @ main()
 2 #include "ARDUINO.h"
 3
 4 int main() {
 5
 6     while (SDL_PollEvent(&e)) {
 7         if (e.type == SDL_KEYDOWN) {
 8
 9             break;
10
11         }
12
13     }
14
15     case SDL_UP: //arrow down button (rotary encoder up)
16     if (currentScreen == TASK_LIST) {
17
18         taskList.nextTask();
19
20     } else if (currentScreen == TASK_PREVIEW) || currentScreen == TASK_RUNNING || currentScreen == TASK_PAUSED) {
21
22         lv_obj_scroll_by(taskScreen.getDescriptionContainer(), 0, 20, LV_ANIM_ON);
23
24     }
25
26     break;
27
28     case SDL_DOWN: //arrow down button (rotary encoder down)
29     if (currentScreen == TASK_LIST) {
30
31         taskList.nextTask();
32
33     } else if (currentScreen == TASK_RUNNING || currentScreen == TASK_PAUSED) {
34
35         lv_obj_scroll_by(taskScreen.getDescriptionContainer(), 0, -20, LV_ANIM_ON);
36
37     }
38
39     break;
40
41     case SDL_RETURN:
42
43     if (currentScreen == TASK_LIST) {
44
45         lv_text_set_text(taskList.getSelectedTitle(), taskList.getSelectedDescription());
46
47         taskScreen.bindTaskList(taskList, taskList.getSelectedIndex());
48
49         taskScreen.bindTaskList(taskList, taskList.getSelectedIndex());
50
51         currentScreen = TASK_RUNNING;
52
53     } else if (currentScreen == TASK_PAUSED) {
54
55         // without resetting elapsed so preview shows remaining time
56
57         taskScreen.bindTaskList(taskList, taskList.getSelectedIndex());
58
59         currentScreen = TASK_PREVIEW;
60
61     } else if (currentScreen == TASK_PREVIEW) {
62
63         lv_text_set_text(taskUI_lv_main_desc, "From preview");
64
65         lv_obj_clear(lv_text_act);
66
67         taskUI_lv_main_cont();
68
69         taskUI_lv_home_screen();
70
71         currentScreen = HOME;
72
73     }
74
75     break;
76
77 }
```

# UW NanoRobotics Group

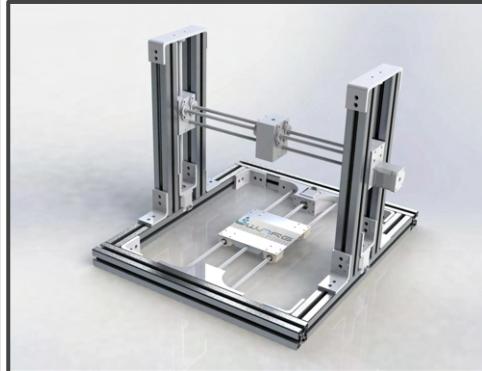
**CHIP Project: Developing a conductive hydrodynamic ink printer that uses nanoparticle inks to fabricate PCBs.**

*PSU Encasing, 5 U-Bracket custom mounts & mechanical drawings, Electronics assembly/validation/measurements, Fluid integration mechanics schematics.*

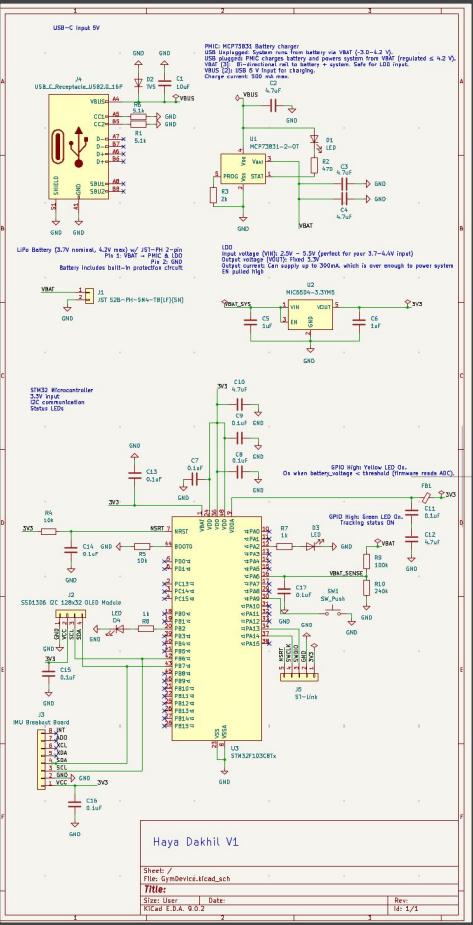
**Tools:** UW Student Machine Shop, Solidworks, 3D printing, DMM, stepper motor, soldering.



## Printer demo



# Fitness Rep Tracker - Hardware (Solo Project)

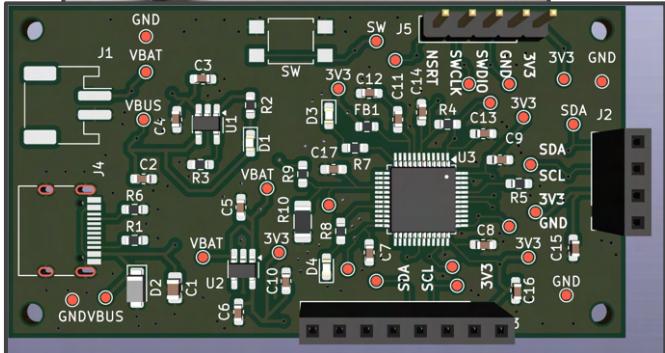
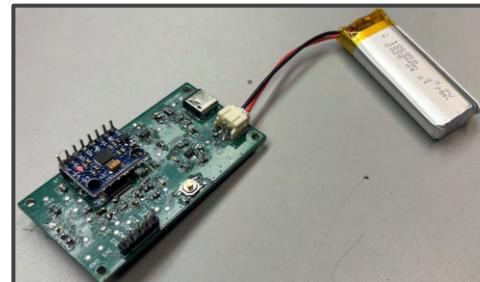
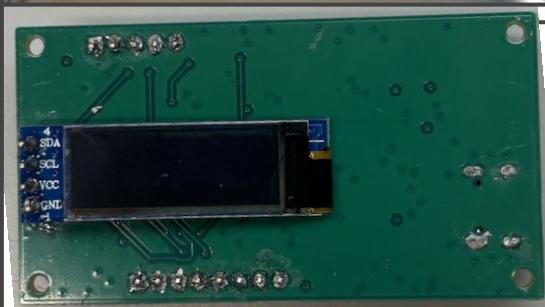
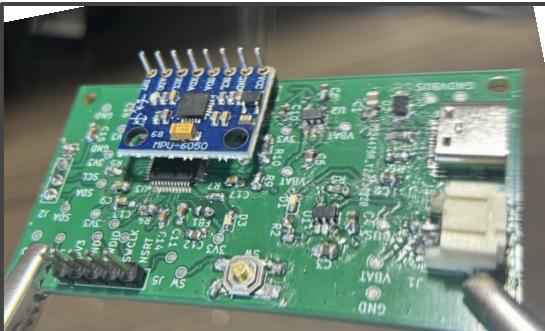


Wrist-mounted STM32 controller board to detect and classify gym exercises with real-time OLED feedback.

*Schematic & PCB design, component selection & BOM, assembly, bring-up, voltage division.*

**Hardware includes:** STM32F103 MCU, MPU-6050 IMU (I<sup>2</sup>C), SSD1306 OLED (I<sup>2</sup>C), 3.3 V LDO regulation, Li-Ion charging via USB-C, low-battery sense divider, status LEDs, ESD protection, SWD programming/debugging header.

**EE software / tools:** KiCad, EE lab tools (oscilloscope, function generator, DMM, power supply, soldering).



# Fitness Rep Tracker - Firmware

Wrist-mounted STM32 controller board to detect and classify gym exercises with real-time OLED feedback.

*Driver development (I<sup>2</sup>C, UART, SysTick), real-time filtering, calibration routines, adaptive thresholding, finite-state machines, custom OLED UI.*

**Firmware includes:** IMU (MPU6050) driver with calibration, OLED (SSD1306) display driver, rolling buffer stats, low-pass filters, rep detection with peak/refractory logic, exercise-specific configs, state machine (boot → select → calibrate → run), UART logging, systick timing, and peripheral init with fallback/error handling.

**Software / Frameworks:** C (PlatformIO), STM32 HAL, FreeRTOS-style FSM (bare-metal), I<sup>2</sup>C, UART, SDL (sim testing).

Adaptive rep detection & rolling statistics

```
src > sensing > C rep_detect.c
88 /**
89  * void rep_detect_end_calibration(exercise_t ex, float *out_mu, float *out_sigma)
90  * {
91  *     if (ex >= EX_COUNT || calib_count[ex] == 0) return;
92  *
93  *     //Compute mean and standard deviation
94  *     float mu = calib_sum[ex] / calib_count[ex];
95  *     float variance = (calib_sum_sq[ex] / calib_count[ex]) - (mu * mu);
96  *     float sigma = sqrtf(fmaxf(variance, 0.0f));
97  *
98  *     //store in runtime context
99  *     REP_CTX[ex].baseline_mu = mu;
100 *     REP_CTX[ex].baseline_sigma = fmaxf(sigma, MIN_SIGMA_FLOOR_G);
101 *     REP_CTX[ex].calibrated = true;
102 *
103 *     //output values
104 *     if (out_mu) *out_mu = mu;
105 *     if (out_sigma) *out_sigma = sigma;
106 * }
107 *
108 * static void update_rolling_stats(exercise_t ex, float new_sample)
109 * {
110 *     if (ex >= EX_COUNT) return;
111 *
112 *     //Add new sample to buffer
113 *     sample_buffer[ex][buffer_index[ex]] = new_sample;
114 *     buffer_index[ex] = (buffer_index[ex] + 1) % ROLLING_BUFFER_SIZE;
115 *
116 *     if (buffer_index[ex] == 0)
117 *     {
118 *         buffer_filled[ex] = true;
119 *     }
120 *
121 *     //compute mean
122 *     float sum = 0.0f;
123 *     uint16_t count = buffer_filled[ex] ? ROLLING_BUFFER_SIZE : buffer_index[ex];
124 *
125 *     for (uint16_t i = 0; i < count; i++)
126 *     {
127 *         sum += sample_buffer[ex][i];
128 *     }
129 *     rep_state[ex].mean = sum / count;
130 *
131 *     //compute standard deviation
132 * }
```

TAB to next move ↵

I<sup>2</sup>C bus initialization with fallback protection

```
src > drivers > C i2c_bus.c
1 /**
2  * @include "i2c_bus.h"
3  * @include "mcu_pinmap.h"
4  */
5
6 HAL_StatusTypeDef i2c_bus_init(void)
7 {
8     hi2c1.Instance = I2C1;
9     hi2c1.Init.ClockSpeed = I2C_BUS_SPEED_FAST_MODE;
10    hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
11    hi2c1.Init.OmAddress1 = I2C_ADDRESS;
12    hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
13    hi2c1.Init.DuaAddressMode = I2C_DUALADDRESS_DISABLE;
14    hi2c1.Init.OmAddress2 = 0;
15    hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
16    hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
17
18    if (HAL_I2C_Init(&hi2c1) != HAL_OK)
19    {
20        //fallback to standard mode if fast mode fails
21        hi2c1.Init.ClockSpeed = I2C_BUS_SPEED_STANDARD_MODE;
22        if (HAL_I2C_Init(&hi2c1) != HAL_OK)
23        {
24            return HAL_ERROR;
25        }
26    }
27    return HAL_OK;
28
29
30 HAL_StatusTypeDef i2c_mem_read(uint16_t dev_address, uint16_t reg_address, uint8_t *pdata, uint16_t Size)
31 {
32     return HAL_I2C_Mem_Read(&hi2c1, dev_address, reg_address, I2C_MEMADD_SIZE_8BIT, pdata, Size, HAL_MAX_DELAY);
33 }
34
35 /*@{ TAB to jump here
36  * @brief Writes a sequence of bytes to a device's internal register.
37  */
38 HAL_StatusTypeDef i2c_mem_write(uint16_t dev_address, uint16_t reg_address, uint8_t *pdata, uint16_t Size)
39 {
40     return HAL_I2C_Mem_Write(&hi2c1, dev_address, reg_address, I2C_MEMADD_SIZE_8BIT, pdata, Size, HAL_MAX_DELAY);
41 }
42
43 }
```

MPU6050 IMU Initialization & Configuration

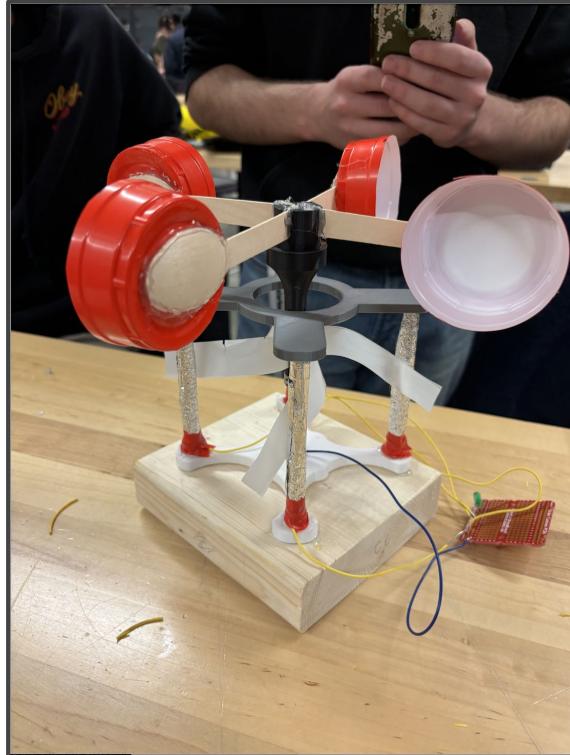
```
src > drivers > C mpu6050.c
23 //Initializes calibration data
24 static float accel_bias[3] = {0.0f, 0.0f, 0.0f};
25 static float gyro_bias[3] = {0.0f, 0.0f, 0.0f};
26
27 //writes a single byte to an mpu register.
28 static HAL_StatusTypeDef MPU6050_WriteRegister(uint8_t reg, uint8_t value)
29 {
30     return I2C_Mem_Write(MPU6050_I2C_ADDR, reg, &value, 1);
31 }
32
33 //reads a single byte from an mpu register.
34 static HAL_StatusTypeDef MPU6050_ReadRegister(uint8_t reg, uint8_t *value)
35 {
36     return I2C_Mem_Read(MPU6050_I2C_ADDR, reg, value, 1);
37 }
38
39 // HAL to ch4: WHO_AM_I generate
40 // initializes MPU
41 HAL_StatusTypeDef mpu6050_init(void)
42 {
43     uint8_t who_am_i;
44     if (MPU6050_ReadRegister(0x75, &who_am_i) != HAL_OK || who_am_i != 0x68)
45     {
46         // LOG("MPU6050 not found or WHO_AM_I mismatched");
47         return HAL_ERROR;
48     }
49
50     // Wake up MPU-6050
51     if (MPU6050_WriteRegister(MPU6050_PWR_MGMT_1, 0x00) != HAL_OK) return HAL_ERROR;
52
53     // set sample rate to I2M_SAMPLE_HZ with sample rate being gyroscope output rate / 1 + SMPLRT_DIV
54     if (MPU6050_WriteRegister(MPU6050_PWR_MGMT_1, 0x00) != HAL_OK) return HAL_ERROR;
55     // gyro output rate = 10Hz (when DLPF is enabled and set to 42Hz or less)
56     // I2M_SAMPLE_HZ = 1000 / 42 = 23.81Hz
57     if (MPU6050_WriteRegister(MPU6050_SMPLRT_DIV, (1000 / I2M_SAMPLE_HZ) - 1) != HAL_OK) return HAL_ERROR;
58
59     // configure DLPF (Digital Low Pass Filter)
60     if (F_EXT_SYNC_SET == 0, DLPF_CFG = 3 (42 Hz) for both accel and gyro
61     if (MPU6050_WriteRegister(MPU6050_CONFIG, 0x03) != HAL_OK) return HAL_ERROR;
62
63     //Configure gyroscope: +/- 250 deg/s (FS_SEL = 0)
64     if (MPU6050_WriteRegister(MPU6050_GYRO_CONFIG, 0x00) != HAL_OK) return HAL_ERROR;
65     GYRO_SCALE_FACTOR = 131.0f; // 131 LSB/deg/s for +/- 250 deg/s
66
67     // configure accelerometer: +/- 2g (APD_SEL = 0)
68 }
```

# Triboelectric Nano-Generator (TENG)

Won 1st place out of 144 students in the TENG prototyping competition to harvest mechanical energy into electricity.

*Solidworks, 3D printing, assembly, materials science, rapid prototyping, cost efficiency*

Tools: Soldering, prototyping / building tools (wood block, paper cups, teflon, aluminum, wooden sticks)



[Video Link Here](#)



# Bio-Mechatronics Design Team

Developing affordable exoskeleton prosthetics for children with muscular dystrophy.

*Base designing & prototyping (ESP32 / servo housing), component / material research, exoskeleton assembly.*

Tools: EMG electrodes, Solidworks, 3D printing, servo motors.

