MKS – ŠABLONY

* **Ovládání LED – low level**

LL\_GPIO\_SetOutputPin(LD1\_GPIO\_Port, LD1\_Pin);

LL\_GPIO\_ResetOutputPin(LD1\_GPIO\_Port, LD1\_Pin);

LL\_GPIO\_TogglePin(LED1\_GPIO\_Port, LED1\_Pin);

* **Ovládání LED – high level**

HAL\_GPIO\_WritePin(LED1\_GPIO\_Port, LED1\_Pin, 1);

HAL\_GPIO\_WritePin(LED1\_GPIO\_Port, LED1\_Pin, 0);

HAL\_GPIO\_WritePin(LED1\_GPIO\_Port, LED1\_Pin, GPIO\_PIN\_RESET);

* **Použití tlačítka – low level**

Porovnání předchozího stavu před stisknutím a stavu v okamžiku stisknutí:

void tlacitko(void)

{

static uint32\_t delay;

static uint32\_t old\_s2;

static uint32\_t old\_s1;

static uint32\_t off\_time;

uint32\_t new\_s2 = LL\_GPIO\_IsInputPinSet(S2\_GPIO\_Port, S2\_Pin);

uint32\_t new\_s1 = LL\_GPIO\_IsInputPinSet(S1\_GPIO\_Port, S1\_Pin);

if (Tick > delay + 40) {

if (old\_s2 && !new\_s2) { // falling edge

off\_time = Tick + LED\_TIME\_SHORT;

LL\_GPIO\_SetOutputPin(LED2\_GPIO\_Port, LED2\_Pin);

}

old\_s2 = new\_s2;

if (old\_s1 && !new\_s1) { // falling edge

off\_time = Tick + LED\_TIME\_LONG;

LL\_GPIO\_SetOutputPin(LED2\_GPIO\_Port, LED2\_Pin);

}

old\_s1 = new\_s1;

if (Tick > off\_time) {

LL\_GPIO\_ResetOutputPin(LED2\_GPIO\_Port, LED2\_Pin);

}

delay = Tick;

}}

* **Použití tlačítka – low level včetně potlačení zákmitů**

void tlacitko(void)

{

static uint16\_t debounce = 0xFFFF;

static uint32\_t delay;

static uint32\_t off\_time;

uint32\_t s1 = LL\_GPIO\_IsInputPinSet(S1\_GPIO\_Port, S1\_Pin);

if (Tick > delay + 5) {

debounce <<= 1;

if (s1 != 0) {

debounce |= 0x0001 ;

if (debounce == 0x7FFF){

off\_time = Tick + LED\_TIME\_LONG;

LL\_GPIO\_SetOutputPin(LED2\_GPIO\_Port, LED2\_Pin);

}

}

if (Tick > off\_time) {

LL\_GPIO\_ResetOutputPin(LED2\_GPIO\_Port, LED2\_Pin);

}

delay = Tick;

}

* **Použití tlačítka – high level**

uint32\_t s2 = HAL\_GPIO\_ReadPin(S2\_GPIO\_Port, S2\_Pin);

uint32\_t s1 = HAL\_GPIO\_ReadPin(S1\_GPIO\_Port, S1\_Pin);

* **Enkodér**

Získání hodnoty z enkodéru:

HAL\_TIM\_Encoder\_Start(&htim1, htim1.Channel);

* **Stavový automat / switch – case šablona**

static enum { SHOW\_POT, SHOW\_VOLT, SHOW\_TEMP } state = SHOW\_POT;

switch (state){

case SHOW\_POT:

…

state = SHOW\_POT;

case SHOW\_VOLT:

…

state = SHOW\_POT;

}

* **If – else šablona**

if (((pole >> i) & 1) == 1) {

…

} else {

…

}

* **For šablona**

for (int8\_t i = 0; i >10 0; i++) {

…

}