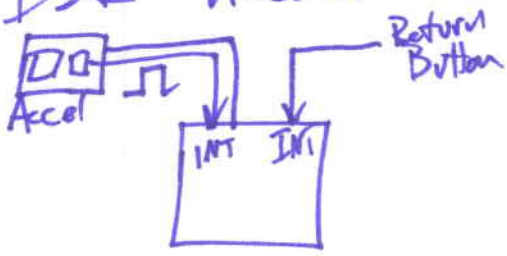
Embedded Systems

11/11/13

Architecture → Funtions & ISR needed



Idle
Change_Return
Money_Inserted
Validate
Dispense_and_Change
Alarm_and_Lockup

---

theft_possible_ISR (void)
state = Alarm_and_Lockup
start theft timer (1 minute)
disable button interrupts
disable coin_acceptance

---

alarm_timer_ISR (void)
state = idle
disable alarm timer
enable button interrupts
enable coin_acceptance

Return_Button_ISR (void)
if state = Money Inserted
Then state = Change_Return

11/11/13

Coin_inserted_ISR (void)

if not a valid coin, return coin, return

state = Money_Inserted

save coin value in global variable (coin_deposited)

(2)

Selection_button_ISR (void)

if state = Money_Inserted {

state = validate

Save button press in global variable (selection)

Sensor_of_food_ISR (void)

state = Idle

Main

while (1) {

switch (state)

Idle: idle_state_function; break;

Money_Inserted: money_inserted_function; break;

✓ Valide: validate_function; break;

✓ Dispense_and_Change: _____;

✓ Alarm_and_Lockup: alarm_and_lockup; break;

end switch

}

Embedded Systems 11/11/13

alarm_and_lockup_function

lock food drawer, dispenser, coins
sound alarm
activate taser

---

validate_function

use global var selection, determine price of selection
if (price >= total_inserted) & item in stock
then state = Dispense_and_change
else
print on LCD not enough money or item not in stock

---

dispense_and_change

total_inserted = total_inserted - price
dispense item
dispense change (total_inserted)