Group 7

Members:

Olga Sharma

Upeksha Sanjeewani

Lihini Hewage

PILL DISPENSER

Project Report

Second-year IOT Project
School of ICT
Metropolia University of Applied Sciences
17 December 2024

Contents

1 Introduction

The Pill Dispenser Project is implemented during the second year first semester at Metropolia University of Applied Sciences. The aim of the project is to create an automated system for dispensing daily medications to patients in a structured and reliable manner. The system is designed to operate autonomously, ensuring medication adherence and providing status updates via a LoRaWAN network.

1.1 Goals of the project

The primary objective of this project is to enhance medication management through automation.
The system is intended to:

- Dispense daily medication at a predefined time using a precise and consistent mechanism.
- Confirm the dispensing process using a piezoelectric sensor, ensuring accuracy and reliability.
- Maintain and persist device states and pill dispensing logs in non-volatile memory for continuity across restarts.
- Communicate real-time device statuses to a central server, enhancing monitoring capabilities.

1.2 Added values

The Pill Dispenser system addresses common challenges in medication adherence and healthcare efficiency.

- **Improved Medication Adherence:** Help maintain consistent medication schedules minimizing missing doses.
- **Enhanced Quality of Life:** Make it easier for patients and caregivers by automated medication dispensing.
- **Healthcare Efficiency:** LoRaWAN integration enables remote monitoring and patient compliance tracking.
- **Error Reduction:** Automated calibration and dispensing minimize human errors, elevating the performance of the device to a higher level.

The project's innovative approach leverages advanced technology to tackle a critical aspect of healthcare, making it an essential tool for both individual and institutional applications.

2 Theoretical Background

A Pill Dispenser is a smart device that helps people take their medicine correctly, especially for chronic conditions like diabetes, hypertension, and cardiovascular diseases. When people don't take their medications properly, it can cause serious health problems and increase hospital visits.

The device uses a mechanical system with a stepper motor and a wheel divided into compartments to release pills accurately. Two key sensors make it work well, a piezoelectric sensor checks if pills are successfully dispensed, and an optical sensor helps position the wheel precisely.

It communicates using LoRaWAN technology, which is great for healthcare because it can send information over long distances while using very little power. This means doctors/nurses can keep track of multiple devices and see if patients are taking their medicines as they should.

Embedded systems are the heart of the Pill Dispenser, combining hardware and software to work smoothly. Non-volatile memory is crucial because it keeps important information like pill counts and dispensing logs safe, even if the power goes out.

By automating medication management, this device does more than just dispense pills. It reduces the mental stress of tracking medications for patients and caregivers. The technology helps prevent mistakes and makes healthcare more efficient, ultimately leading to better health outcomes.

3 Methods and Material

The following chapter is dedicated to the explanation of the materials utilized for the project. Firstly, the topic is introduced from a hardware point of view in the first subchapter. Secondly, the software aspect is discussed in the second subchapter along with the applied methods.

3.1 Hardware Requirements

For the development of this project, the components in the below Figure 1 work together to create an automated medication management system that can reliably dispense pills while communicating status updates remotely. The combination of electronics, mechanics, and wireless connectivity enables enhanced medication adherence for patients.

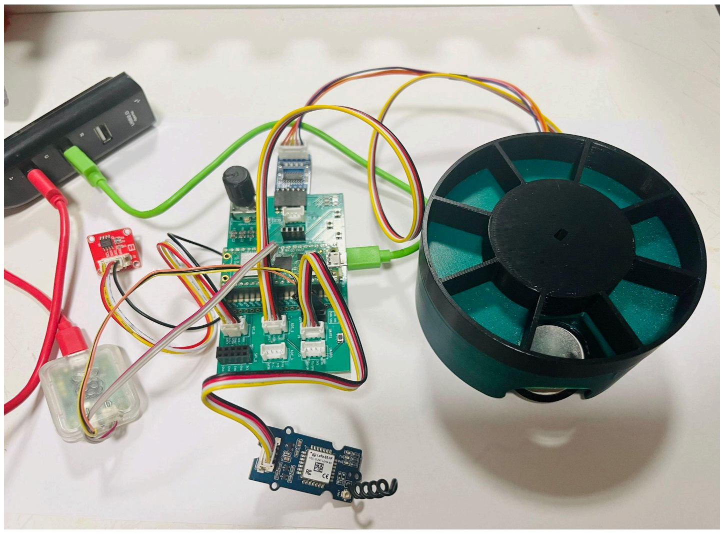
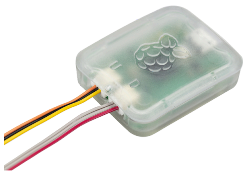


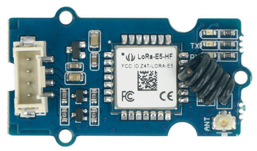
Figure 1: Pill Dispenser

Let's look into each of the above components separately,



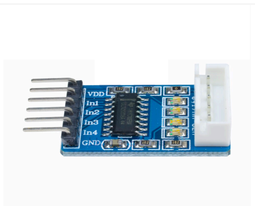
- **PicoProbe debugger:** The Raspberry Pi Debug Probe is a USB device that provides both a UART serial port and a standard Arm Serial Wire Debug (SWD) interface. The probe is designed for easy, solderless, plug-and-play debugging.[1].

Figure 2: Raspberry Pi Debug Probe [2]



- **LoRaWAN module:** This wireless module enables long-range, low-power communication using the LoRaWAN protocol, allowing the device to connect to remote monitoring systems.

Figure 3: Grove, LoRa-E5 [3]



- **Stepper motor driver:** This specialized component controls the precise rotation of the stepper motor that powers the medication dispenser wheel.

Figure 4: Stepper Motor [6]

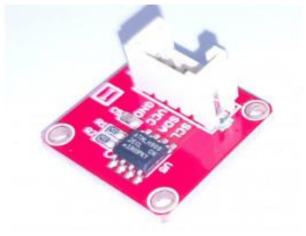


Figure 5: I2C EEPROM [4]

- **I2C EEPROM:** This non-volatile memory chip stores important data like the device's configuration, medication schedules, and usage logs, even when the power is off.

Figure 6: Raspberry pi pico [5]

- **Controller PCB:** This is the main printed circuit board that houses the microcontroller and integrates all the electronic components to coordinate the dispenser's operation.

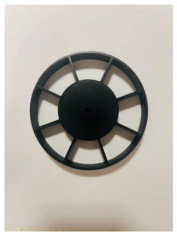


Figure 7: Dispenser wheel

- **Dispenser wheel:** The dispenser wheel is the component that holds the pills and rotates to align the compartments with the drop tube. The wheel has eight compartments total - seven for holding pills and one for the calibration opening. The calibration opening is detected by the optical sensor in the dispenser base, allowing the system to precisely align the wheel. As the wheel rotates, the piezo sensor in the dispenser base monitors for the vibrations or pressure changes that indicate a pill has been successfully dispensed from the aligned compartment.

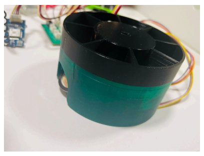


Figure 8: Dispenser base

- **Dispenser base:** This is the structural foundation that holds and supports the dispenser wheel mechanism.

- **Stepper motor shaft:** This refers to the rotating shaft that is driven by a stepper motor, which is a type of electric motor that moves in discrete steps.

Figure 7: Stepper Motor

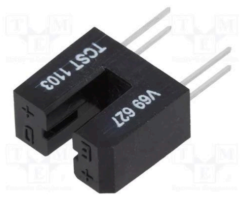


Figure 8: Opto fork [7]

- **Optical sensor (opto fork):** The optical sensor, or opto fork, is used to detect the calibration opening on the dispenser wheel. When the opening aligns with the sensor, it registers a change in the sensor's output, allowing the system to determine the exact position of the wheel. This is crucial for accurately dispensing pills from the correct compartment.

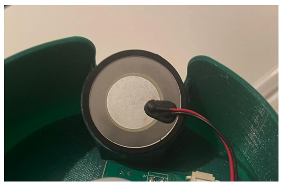


Figure 9: Piezo sensor

- **Piezo sensor:** The piezo sensor is used to detect when a pill has been successfully dispensed. When a pill falls through the drop tube and hits the piezo sensor, it creates a detectable vibration or change in pressure, which is registered by the controller PCB. This feedback allows the system to confirm that a pill has been dispensed.

The overall device is a positioning or control mechanism that uses a combination of a stepper motor, optical sensors, and piezoelectric sensors to precisely control and monitor the movement of the shaft.

3.2 Software Requirements and Methods

The pill dispenser system is being implemented by using CLion IDE and the programming language used was C language.

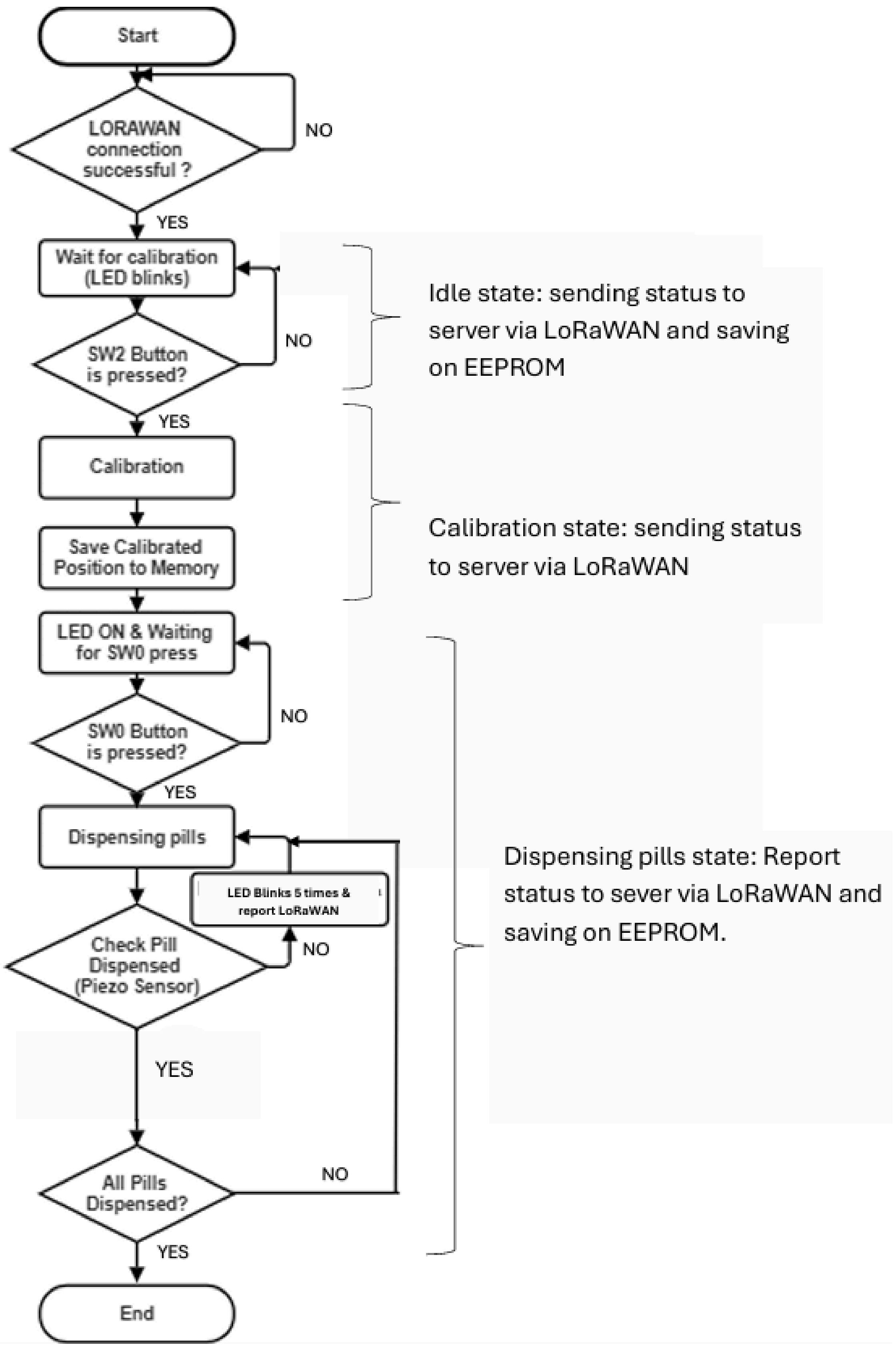
4 Implementation

4.1 Operating principles of the software

The software implemented that is running on the controller PCB is responsible for coordinating the various components and implementing the core functionality of the pill dispenser. The main software operating principles include functions as follows.

- **Calibration:**
  When the device is powered on, the software waits for a button press to initiate the calibration process. It then turns the dispenser wheel at least one full rotation, using the optical sensor to detect the falling edge and align the wheel to a known starting position. This calibration ensures the system knows the exact position of the wheel and can accurately dispense pills from the correct compartment.
- **Pill Dispensing:**
  After calibration, the software polls until a button press to begin the regular pill dispensing cycle. It then rotates the dispenser wheel in increments, pausing after each movement. If during the movement the piezo sensor was triggered, that indicates a pill has been successfully dispensed. If no pill is detected, a LED blinks five times to indicate a failed dispense. The wheel continues to rotate through the 7 pill compartments, dispensing the daily medication.
- **State Management:**
  The software maintains the current state of the pill dispenser, including the number of pills remaining and dispensing history. This information is stored in non-volatile memory (EEPROM) so that it persists across power cycles and reboots.
- **LoRaWAN Communication:**
  The software connects to the LoRaWAN to report the current state of the pill dispenser to a remote server. It handles the LoRaWAN networking protocols, including joining the network, sending status updates, and processing any commands received from the server.
- **Error Handling:**
  The software handles the potential error issues, such as power loss during any state of the system. It implements recovery mechanisms to recalibrate the system and resume normal operation without losing any pills.

4.2 Flow Chart


Start
LORAWAN connection successful ?
NO
YES
Wait for calibration (LED blinks)
SW2 Button is pressed?
NO
YES
Calibration
Save Calibrated Position to Memory
LED ON & Waiting for SW0 press
SW0 Button is pressed?
NO
YES
Dispensing pills
LED Blinks 5 times & report LoRaWAN
Check Pill Dispensed (Piezo Sensor)
NO
YES
All Pills Dispensed?
NO
YES
End

Idle state: sending status to server via LoRaWAN and saving on EEPROM

Calibration state: sending status to server via LoRaWAN

Dispensing pills state: Report status to sever via LoRaWAN and saving on EEPROM.

4.3 Software algorithms

Firstly, to execute the programme successfully, the required libraries are defined. Below table 1 there is an explanation of each library.

Library	Description
<stdbool.h>	Provides boolean type and values (true/false)
<stdio.h>	Standard input/output library for basic I/O operations
<string.h>	Provides string manipulation functions
<time.h>	Provides time-related functions and types
"pico/stdlib.h"	Raspberry Pi Pico standard library for basic microcontroller functions
"hardware/gpio.h"	GPIO (General Purpose Input/Output) control functions for Raspberry Pi Pico
"hardware/i2c.h"	I2C (Inter-Integrated Circuit) communication functions
"pico/util/queue.h"	Utility library for queue data structures
"hardware/uart.h"	UART (Universal Asynchronous Receiver/Transmitter) communication functions

Table 1: The libraries of the programme

Secondly, the pins related to LEDs, switches, motor, opto_fork, piezo sensor, and LoRa communication are defined in the programme.

In addition, the programme contains overall 27 functions and the main programme. The former is described in detail in Chapter 4.3.1 and the latter one in Chapter 4.3.2.

4.3.1 Functions

As mentioned prior, the programme consists of overall 27 functions, which are listed in Table 2. The table defines the Function's name and the taken argument in the first column and explains their purpose in the second column.

Function Name	Description
setup_UART()	Initializes UART communication for LoRa module with specified pins and baud rate
send_command_LoRa()	Sends a command to the LoRa module via UART and checks for a response
read_response_with_us()	Reads a response from UART within a specified microsecond timeout, storing it in a buffer

process_response_status()	Checks if the LoRa module response contains "OK"
i2c_setup()	Initializes I2C communication with specified pins and baud rate
eeprom_write_system_state()	Writes data to EEPROM with an address, including an inverse data check
eeprom_read_system_state()	Reads data from EEPROM at a specific address, returning a struct with value and inverse value
value_is_valid()	Checks if the read EEPROM value is valid by comparing it with its inverse
piezo_interrupt()	Interrupt handler for the piezo sensor, sets a flag when triggered
SetMotor()	Initializes GPIO pins for stepper motor control
SetOptoFork()	Initializes the opto fork sensor GPIO pin
SetPiezoSensor()	Initializes the piezo sensor GPIO pin
activate_motor_step()	Performs a single step of the stepper motor in forward direction
activate_motor_step_reverse()	Performs a single step of the stepper motor in reverse direction
calibrate_motor()	Calibrates the stepper motor using the opto fork sensor
run_motor()	Runs the stepper motor for a specified number of rotations
recalib_motor()	Recalibrates the motor based on stored days passed
IdleMode()	Blinks the LED to indicate idle state
Calibrating()	Handles the calibration process and sends a message to the server
DispensingPills()	Manages the pill dispensing process, tracking days and using the piezo sensor
HandleError()	Blinks the LED to indicate an error condition
SetupButtons()	Initializes GPIO pins for buttons
SetupLed()	Initializes the LED GPIO pin
connect_LORA()	Attempts to connect to the LoRa module
handle_join_command()	Handles the LoRa module join process with the server
handle_msg_command()	Sends a message to the server via LoRa module
send_connection_command_server()	Configures and establishes connection with the server
main()	Main program loop managing the state machine of the pill dispenser

Table 2: The functions of the programme

4.3.2 Main Programme

The main() function serves as the central control logic for the pill dispenser system. It initializes all hardware components (motors, sensors, communication interfaces) and sets up the system's initial state by reading the previous state from EEPROM.
The function implements a state machine with three primary states:

1. STATE_WAIT: System is in idle mode, waiting for calibration
2. STATE_CALIBRATION: Handles the motor calibration process
3. STATE_DISPENSE_PILLS: Manages the pill dispensing sequence

The function first checks the previous system state stored in EEPROM to determine the appropriate initial state. This allows the system to recover from unexpected power interruptions such as follows.

- If the system was interrupted during calibration, it returns to the idle state.
- If the system was interrupted during pill dispensing, it can resume or recalibrate.
- If the system is in a clean state, it proceeds accordingly.

The main loop continuously monitors button presses and transitions between states, managing the entire pill dispensing workflow. It also handles server communication via LoRa, sending status messages at different stages of the process.
This design ensures robust operation, state persistence, and the ability to recover from unexpected interruptions.

5 Team work Allocation

Upeksha Samarawickrama Liyanage:

Hardware initialization / Motor control (initialization and implement functions):
SetMotor (), SetOptoFork (), SetPiezoSensor (), SetupButtons (), SetupLed (),
activate_motor_step (), run_motor (), calibrate_motor (),
recalib_motor (),activate_motor_step_reverse ()
Reviewed and edited by Olga and Lihini

Lihini Hewage:

LoRaWAN Communication (initialization and implement functions):
connect_LORA (), send_command_LoRa (), handle_join_command (),
handle_msg_command ()
setup_UART (), read_response_with_us (), process_response_status ()
Reviewed and edited by Olga and Upeksha

Olga Sharma:

State machine / EEPROM / Pill dispensing logic / set up LoRa connection / Implement state machine
Handle states: STATE_WAIT, STATE_CALIBRATION, and STATE_DISPENSE_PILLS and main (),
i2c_setup (), eeprom_write_system_state (), eeprom_read_system_state (),
, piezo interrupt(),
IdleMode (), Handle Error (), Calibrating (), DispensingPills (),
send_connection_command_server (), process_response_status ()
Reviewed and edited by Upeksha and Lihini

Common Tasks for All Members:
Test the system
Test LoRaWAN communication
Ensure data integrity in EEPROM.
Documentation:
-Prepare flowchart.
-Final report.

6 Conclusion

The automated pill dispenser system aims to improve patient life quality and healthcare efficiency through reliable, convenient medication management. Key features include precise mechanical control, electronic monitoring, and remote LoRaWAN communication. The modular, fault-tolerant software powers the system's robust functionality, ensuring seamless operation and minimal risk of missing doses. Overall, this innovative solution enhances medication adherence and healthcare delivery.

References

1. Raspberry Pi Documentation. Raspberry Pi Debug Probe. Accessed: 14.12.2024 https://www.raspberrypi.com/documentation/microcontrollers/debug-probe.html

2. Reichelt, Raspberry Pi Debug Probe. Accessed: 14.12.2024 https://www.reichelt.com/fi/en/shop/product/raspberry_pi_debug_probe-343288?PROVID=2818&gad_source=1&gclid=CjwKCAiA9vS6BhA9EiwAJpnXw5sJrDvlzihzbiNAnLcAvS9Fxqf46MMMLcauzBpyqrvv9letY_doPBoC4U4QAvD_BwE

3. Hobbyhall, Grove, LoRa-E5 STM32WLE5JC, EU868 / US915, LoRaWAN-radiomoduuli, Seeedstudio 113020091 https://hobbyhall.fi/fi/tietokoneet-ja-pelaaminen/mikrokontrollerit/grove-lora-e5-stm32wle5jc-eu868-us915-lorawan?id=15957673&mid=19662353&gad_source=1&gclid=CjwKCAiA9vS6BhA9EiwAJpnXww-SD0koq0uqZ6m1hsBBn0jwhdyWh-oVeAlA7Ap_xafJJSImMuWM7xoCBtkQAvD_BwE

4. Partco, Crowtail I2C EEPROM 2.0 https://www.partco.fi/en/diy-kits/crowtail/23675-ect-ct010021e.html

5. Reichelt, Raspberry Pi Pico W, RP2040, Cortex-M0+, WLAN, microUSB. Accessed: 14.12.2024 https://www.reichelt.com/fi/en/shop/product/raspberry_pi_pico_w_rp2040_cortex-m0_wlan_microusb-329646?PROVID=2818&gad_source=1&gclid=CjwKCAiA9vS6BhA9EiwAJpnXw13eixyCUJmHX9upXr-OMNciZ9eJxIwLqJSyICTVXWyiJe2SVlOIbxoCMKQQAvD_BwE

6. Sintosen Palvelut, ULN2003 Stepper Motor Driver. Accessed: 15.12.2024 https://kauppa.sintosen.com/product/1278/?gad_source=1&gclid=CjwKCAiAmfq6BhAsEiwAX1jsZ-mTQtwtWGznSyTxBIXokqg9meUUa9P35nPeTUo-D2-DN6T3HoD6ORoC7RwQAvD_BwE

7. TEM Electronic Components, Sensor: Optocoupler; Transmitter-Receiver (slit); Slit Width: 3.1mm. Accessed:15.12.2024 https://www.tme.eu/fi/details/tcst1103/valosahkoiset-anturit-piirilevylle/vishay/?brutto=1&currency=EUR&utm_source=google&utm_medium=cpc&utm_campaign=FINLANDIA%20[PMAX]&gad_source=1&gclid=CjwKCAiAmfq6BhAsEiwAX1jsZ88bw-F25askQPPqUyBXQlMddd9pFbz3Qa7gfbDYwuWkKDweN9_vgBoCzz0QAvD_BwE