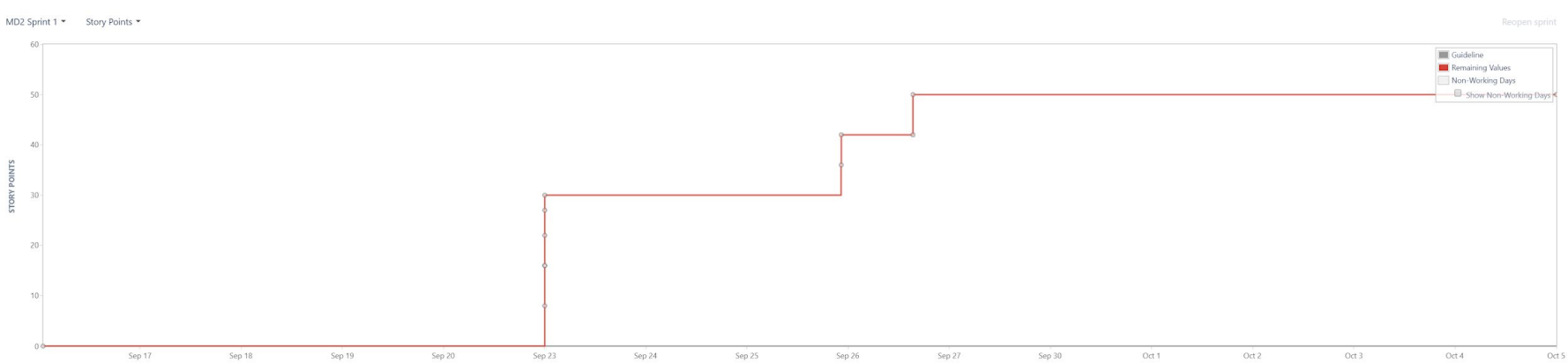
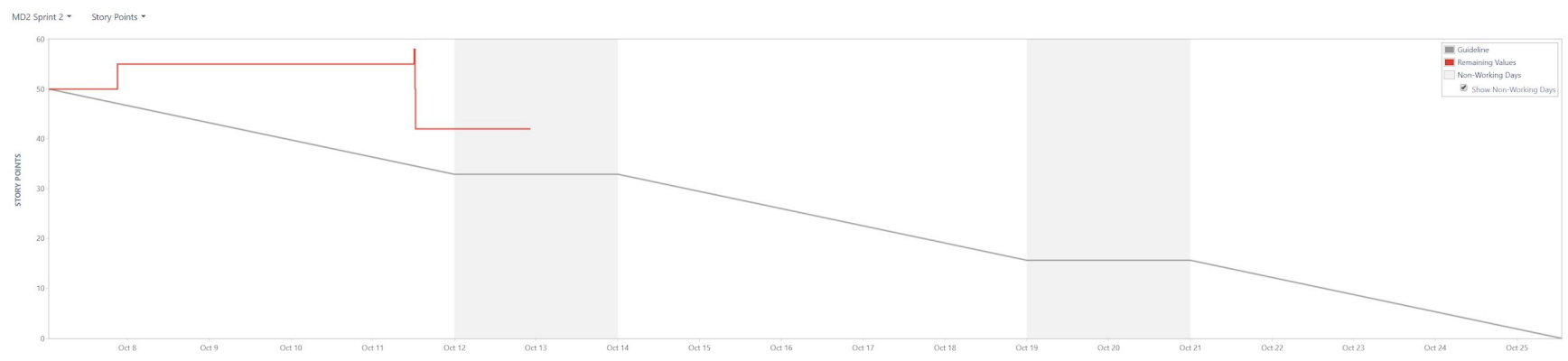
Medical Delivery

IoT - Tussentijdse presentatie

Sprint 1 - Burndown Chart



Sprint 2 - Burndown Chart



Sprint 2 - User story done MD2-14

Medical Delivery #2 / MD2-14

Technical Story: Send data from microcontroller via LoRa module to LoRa gateway

Edit | Comment | Assign | More | To Do | In Progress | Done

Details

Type:	Story	Status:	DONE (View Workflow)
Priority:	Highest	Resolution:	Done
Labels:	None		
Epic Link:	Hardware		
Sprint:	MD2 Sprint 1, MD2 Sprint 2		

Description

Narrative

Connect the microcontroller with the LoRa module. Send some test data over the LoRaWAN network. Since we only have chosen a microcontroller that would theoretically be good for this project, select an appropriate one that is available in the meantime in the labs.

Acceptance Criteria

- Is able to transmit data over the LoRaWAN network.
- Code and connections documented.

Sprint 2 - User story done MD2-17

Medical Delivery #2 / MD2-17

Technical story: 9dof data to microcontroller

Edit | Comment | Assign | More | To Do | In Progress | Done

Details

Type:	Story	Status:	DONE (View Workflow)
Priority:	High	Resolution:	Done
Labels:	None		
Epic Link:	Hardware		
Sprint:	MD2 Sprint 1, MD2 Sprint 2		

Description

Narrative

The 9dof will send data over I^2C to the microcontroller. Logging of the data and handling it. Maybe understanding what part of the data are the different sensors.

Acceptance Criteria

- Logging data from the 9dof onto serial monitor

In Progress

Acceptance Criteria

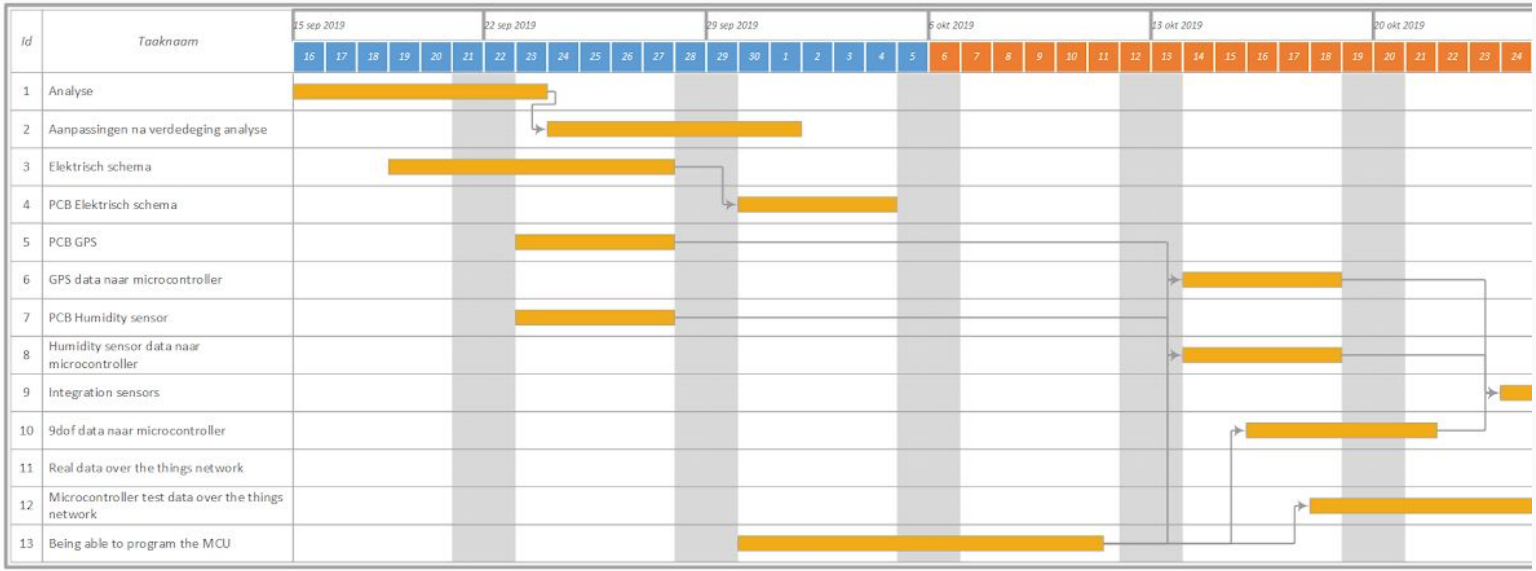
- ✔ The PCB has been designed
- ✘ The components have been assembled
- ✘ The PCB has been tested

User stories meer opdelen in kleinere user stories. Rekening houden met afprinten pcb.

Updates in Analyse

- Added power consumption document
- Added code documentation
- Added sprint meetings doc
- Updated electrical scheme
- Updated flowchart
- Added regio’s in tabel marktonderzoek

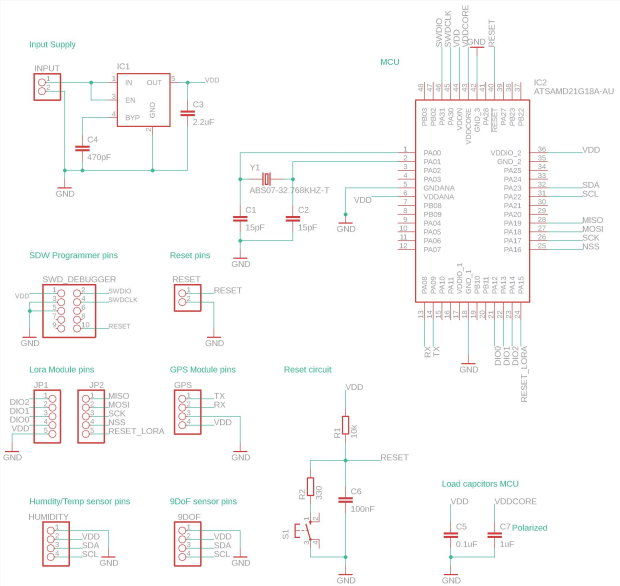
Verdeling taken - Release plan

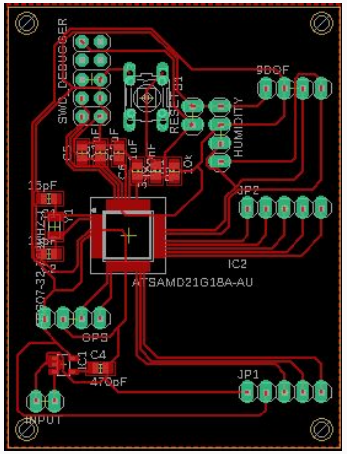


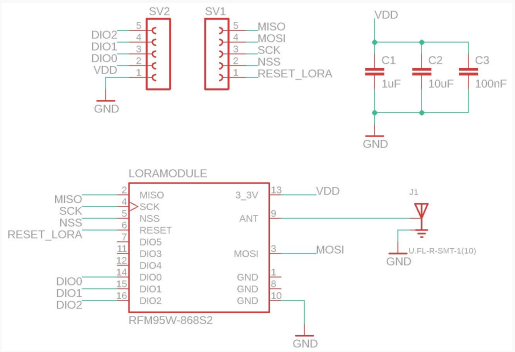
Presteerde werk

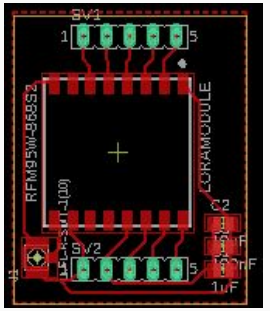
- Analyse geupdate telkens bij aanpassingen
- Documenteren van code
- Denken over integreren van componenten met elkaar
- Componenten dat verkrijgbaar waren in het labo aanpassen in schema’s en pcb designs

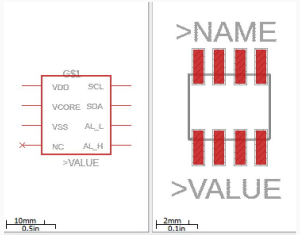
Jarno





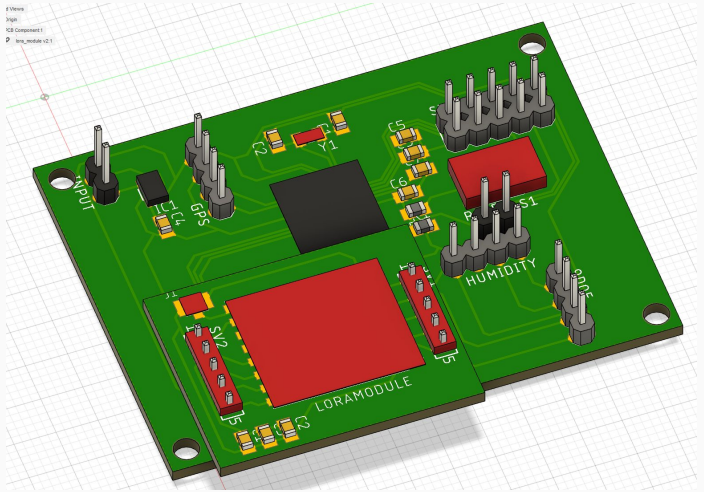




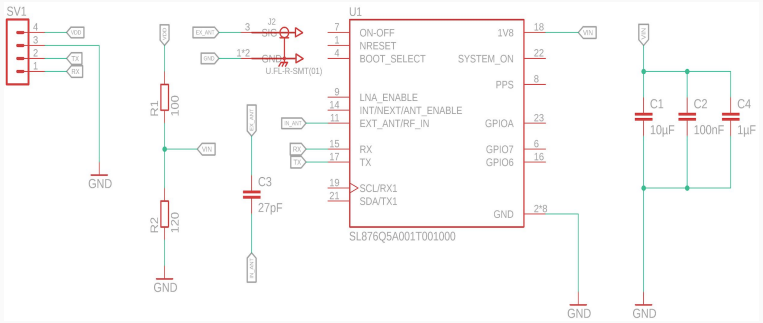


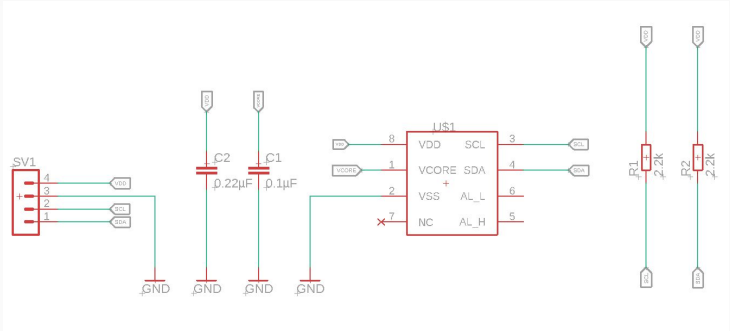
HIH6030-021-001

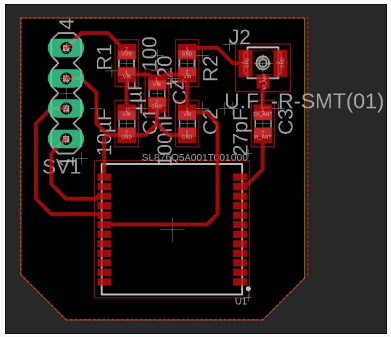
Honeywell HumidIconTM Digital Humidity/Temperature Sensors

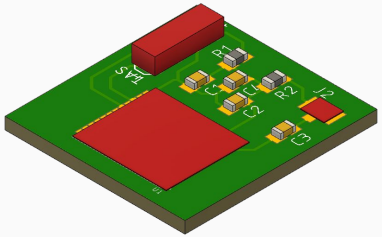


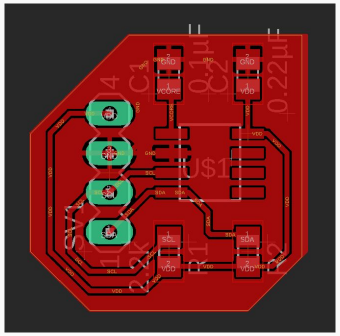
Imad

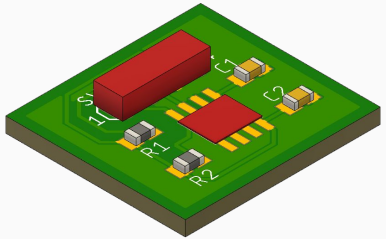

SV1
U1
ON-OFF
NRESET
BOOT_SELECT
LNA_ENABLE
INT/NEXT/ANT_ENABLE
EXT_ANT/RF_IN
RX
TX
SCL/RX1
SDA/TX1
1V8
SYSTEM_ON
PPS
GPIOA
GPIO7
GPIO6
GND
SL876Q5A001T001000
U.FL-R-SMT(01)
R1
100
R2
120
C1
10µF
C2
100nF
C3
27pF
C4
1µF


SV1
U$1
VDD
VCORE
VSS
NC
SCL
SDA
AL_L
AL_H
C1
0.1µF
C2
0.22µF
R1
2.2k
R2
2.2k


J4
R1
100
J2
R2
120
C1
10µF
C2
100nF
C3
27pF
U.FL-R-SMT(01)
SV1
SL876Q5A001T001000







Cristian

```
void printAccel()
{

  // To read from the accelerometer, you must first call the
  // readAccel() function. When this exits, it'll update the
  // ax, ay, and az variables with the most current data.
   if ( imu.accelAvailable() ){
  imu.readAccel();
   }
  float accelX = imu.calcAccel(imu.ax);
  float accelY = imu.calcAccel(imu.ay);
  float accelZ = imu.calcAccel(imu.az);

 // total acceleration
 float acceltot = sqrt(pow(accelX,2)+pow(accelY,2)+pow(accelZ,2));

 float accfilterdX = accelX/acceltot;
 float accfilterdY = accelY/acceltot;
float accfilterdZ = accelY/acceltot;

    if ((lastPrint + PRINT_SPEED) < millis())
  {
  // Now we can use the ax, ay, and az variables as we please.
  Serial.print("X: ");
  Serial.print(accfilterdX);
  Serial.print("Y: ");
  Serial.print(", ");
  Serial.print(accfilterdY);
   Serial.print("Z: ");
  Serial.print(", ");
  Serial.println(accfilterdZ);
  Serial.println();


  }

  delay (500);
}
```

```
void updateGyro() {



  // Update the sensor values whenever new data is available
  if ( imu.gyroAvailable() )
  {
    // update gx, gy, and gz variables with the most current data.
    imu.readGyro();
  }
   if ( imu.accelAvailable() ){
  imu.readAccel();
   }
  float accelX = imu.calcAccel(imu.ax);
  float accelY = imu.calcAccel(imu.ay);
  float accelZ = imu.calcAccel(imu.az);


   //to do add Y AND X BASE TO control value
   //Accelerometer for better values
  float roll = atan2(accelY , accelZ) * 57.3;


  // Serial.print("roll :  ");
  // Serial.println(roll);

  //complementaire filter
  float f_gz = imu.calcGyro(imu.gz);
  float dtC = float(millis())/1000.0;
  a=tau/(tau+dtC);
  rotation = a* (rotation + f_gz * dtC) + (1-a) * (roll);
       Serial.print("rotation :  ");
       Serial.println(abs(rotation));

   if ((abs(rotation) >= 102.0)){
       Serial.println("Box has fallen over");
   }
   }
```

Oussama

ttn-abp

```
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>

// LoRaWAN NwkSKey, network session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const PROGMEM u1_t NWKSKEY[16] = { 0x96, 0x74, 0x6C, 0x57, 0xD8, 0xE3, 0xB1, 0xD6, 0x3A, 0x04, 0x83, 0x04, 0xC4, 0x56, 0x8D, 0x36 };

// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const u1_t PROGMEM APPSKEY[16] = { 0x1F, 0x42, 0x97, 0x73, 0xAE, 0xEA, 0xDA, 0xE8, 0xD4, 0xEE, 0x45, 0xF7, 0x54, 0xD4, 0xAE, 0x57 };

// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR = 0x26011BFE ; // <-- Change this address for every node!

// These callbacks are only used in over-the-air activation, so they are
// left empty here (we cannot leave them out completely unless
// DISABLE_JOIN is set in config.h, otherwise the linker will complain).
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }

static uint8_t mydata[] = "IMAD WAT!";
static osjob_t sendjob;

// Schedule TX every this many seconds (might become longer due to duty
// cycle limitations).
```

Fields

```
{
  "text": "IMAD WAT!"
}
```

Metadata

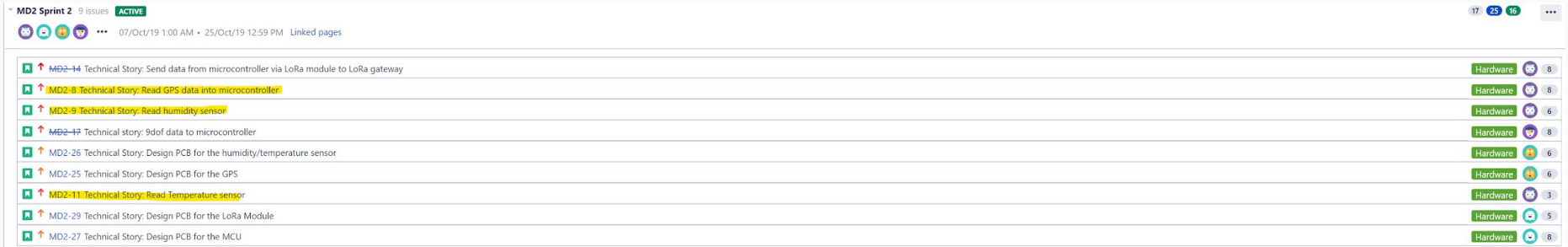
```
{
  "time": "2019-10-11T10:15:27.413983049Z",
  "frequency": 867.9,
  "modulation": "LORA",
  "data_rate": "SF7BW125",
  "coding_rate": "4/5",
  "gateways": [
    {
      "gtw_id": "eui-7276ff002e062e3e",
      "timestamp": 30077299,
      "time": "2019-10-11T10:15:26.329671Z",
      "channel": 4,
      "rssi": -114,
      "snr": -4,
      "latitude": 51.21708,
      "longitude": 4.41928,
      "altitude": 59
    }
  ]
}
```

Estimated Airtime

41.216 ms

```
function Decoder(bytes, port) {
  var result = "";
  for(var i = 0; i < bytes.length; i++){
    result += (String.fromCharCode(bytes[i]));
  }
  return {text: result};
}
```

TODO


MD2 Sprint 2 9 issues ACTIVE
17 25 16
07/Oct/19 1:00 AM • 25/Oct/19 12:59 PM Linked pages
~~MD2-14~~ Technical Story: Send data from microcontroller via LoRa module to LoRa gateway Hardware 8
~~MD2-8~~ Technical Story: Read GPS data into microcontroller Hardware 8
~~MD2-9~~ Technical Story: Read humidity sensor Hardware 6
~~MD2-17~~ Technical story: 9dof data to microcontroller Hardware 8
MD2-26 Technical Story: Design PCB for the humidity/temperature sensor Hardware 6
MD2-25 Technical Story: Design PCB for the GPS Hardware 6
~~MD2-11~~ Technical Story: Read Temperature sensor Hardware 3
MD2-29 Technical Story: Design PCB for the LoRa Module Hardware 5
MD2-27 Technical Story: Design PCB for the MCU Hardware 8