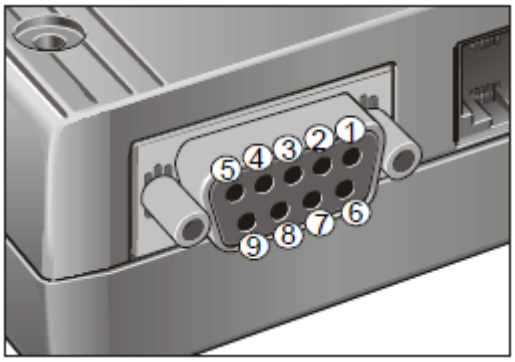
1 Tekniske overvejelser

1.1 RS232-kabel

Forbindelsen til GSM-modemet sker vha. et 9-pins kabel. Det er forbundet til STK-500's Spare port og GSM-modemets RS-232 interface.

Når der skrives til GSM-modemet køres dette igennem UART-driveren, som sender og modtager igennem RS-232 stikket. Interfacet på STK-500 og GSM-modemet er det samme, hvilket resulterer i, at et han-han kabel mellem de to vil få modtagerpindene sat sammen senderpindene sat sammen, hvilket resulterer i ingen kommunikation.

For at løse dette, designede vi et 9-pins kabel, hvor pin 2 og 3 var byttet, således at det passede med senderpindene blev sat til modtagerpindene.



Figur 1: 9-Pin hun stik for STK-500 Spare og GSM-modem

1.2 Drivers

Til projektet er der blevet brugt nogle drivere fra opgaverne fra AMS-forløbet, samt udleveret drivere til UART'en og LED'erne. Herunder er en liste over de drivere, som vi selv har arbejdet på.

1.2.1 LM75

Driveren til temperatursensoren, LM75, er baseret på en opgave fra AMS-forløbet. Den kommunikerer med STK-500 vha. I^2C-bussen og kan i princippet måle fra $-155°C$ til $+155°C$, hvilket vi dog ikke har test eller finder relevant for denne opstilling.

Fra tasken `void LM75SensorTask(void *pvParameters)` kaldes LM75-driverens funktion til, at modtage ny temperatur fra LM75-printet, vist i Listing 1.

Listing 1: LM75's metode til at spørge på ny data

```
1  int LM75_temperature(unsigned char SensorAddress)
2  {
3    i2c_start();
4
5    unsigned char address = ((0b01001000 | SensorAddress) << 1) | 0↩
       x01;
6    i2c_write(address); //Address write
7
8    unsigned char tempMSB = i2c_read(0x00);
9    unsigned char tempLSB = i2c_read(0x01);
10   i2c_stop();
11
12   return (tempLSB>>7) | (tempMSB<<1);
13 }
```

Ved at give en adressen på slaven i I^{2}C-bussen som parameter, startes I^{2}C'en og der sendes en forespørgsel på denne slave, hvorefter denne sender sin data retur. Eftersom temperaturen er 9 bit lang, shiftes de to læsninger, således MSB og LSB står korrekt og returneres i en int.

1.2.2 LCD162

Driveren til DragonKit, der indeholder et LCD162 display, er baseret på en opgave fra AMS-forløbet. Denne kan udskrive tekst på displayet, hvilket benyttes til at vise den aktuelle temperatur.

Der udskrives hovedsagligt strings og ints på displayet. På Listing 2 og Listing 3 ses hvordan disse funktioner er skrevet.

Listing 2: LCD162 metode til visning af en string

```
1  void LCDDispString(char* str)
2  {
3    for(int i = 0 ; i < 32 ; i++)
4    {
5      if(str[i] == '\0')
6        break;
7      sendData(str[i]);
```

```
8     }
9 }
```

Listing 2 virker således, at den modtager en char*, hvor den steppes igennem char for char og udskrives indtil hele strengen er udskrevet, eller har skrevet flere tegn end der kan være på displayet.

Listing 3: LCD162 metode til visning af en integers

```
1 void LCDDispInteger(int i)
2 {
3   char arr[3];
4   itoa(i, arr, 10);
5   LCDDispString(arr);
6 }
```

Listing 3 virker således, at den tager imod den ønskede int, som overføres til et array med plads til 3 chars[1], hvorefter itoa-funktionen skriver dataen in i arrayet i decimal format. Til sidst udskrives det vha. LCDDispString, vist i Listing 2.

1.3 Free RTOS

1.4 Ideel opbygning

[1] Her burde være 6 pladser, så der er plads til -32768 og op til +32767