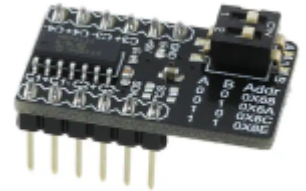




Convertisseur analogique numérique

[Mise à jour le 3/2/2020]



MCP3424

- *Source* : [wiki](#)

Le module DFR0316 est un module de conversion de données à circuit MCP3424 **18-Bit ADC-4 Channel**

- *Distributeur* : [Mouser](#)
- *Caractéristiques*
 - Alimentation: 2,7 à 5,5 Vcc
 - Consommation en standby: 300 nA sous 5 Vcc
 - Gain programmable PGA: x1, x2, x4 ou x8
 - Plage de mesure différentielle: de -2,048/Gain à +2,048/Gain (par exemple de -0,512 à +0,512 V pour un gain de 4)
 - Résolution programmable: 12, 14, 16 ou 18 bits
 - Vitesse: 240, 60, 15 ou 3,75 mesures/seconde
 - Erreur gain: 0,05 % (gain = 1 sous 18 bits)
 - Erreur offset: 15 μ V (gain = 1 sous 18 bits)
 - Interface I2C: 0X68, 0X6A, 0X6C ou 0X6E (sélectionnable par dip-switch)
 - Référence de tension interne: 2,048 Vcc \pm 0,05%
 - T° de service: -40 à +125 °C
 - Dimensions: 27 x 16 x 12 mm



- **Présentation** du circuit sur [Github](#)
- *Documentation* : [pdf à télécharger](#)

- **Programmation d'une carte Arduino Uno R3** 


```
// Initialisation de la console
Serial.begin(9600);
// Initialisation du bus I2C
Wire.begin();

// Enable power for MCP342x (needed for FL100 shield only)
//pinMode(9, OUTPUT);
//digitalWrite(9, HIGH);

// Reset du circuit
MCP342x::generalCallReset();
delay(1); // MC342x nécessite 300us pour se paramétrer, on attend
1ms

// Vérification de la présence du MCP342x
Wire.requestFrom(address, (uint8_t)1);
if (!Wire.available())
{
    Serial.print("Pas de circuit à cette adresse ");
    Serial.println(address, HEX);
    while (1)
        ;
}
}

void loop(void)
{
    long value = 0, tension_mV = 0;
    int gain, resolution;
    MCP342x::Config status;
    // -----
    // -----
    // Déclenchement d'une conversion. La méthode convertAndRead()
attend la fin de la conversion
    // On passe les paramètres suivants à la fonction dans l'ordre :
    // - canal : channel1, channel2, channel3 ou channel4 pour un
MCP3424
    // - type de conversions : oneshot ou continous
    // - résolution : resolution12, resolution14, resolution16 ou
resolution18
    // - gain : gain1, gain2, gain4 ou gain8
    // timeout = temps accordé à la transaction en ms (1000000 dans cet
exemple !)
    // A l'issue de la conversion
    // - value = gain * (2^résolution/2^12) * tension du canal
sélectionné en mv
    // - status identifie la valeur du gain et de la résolution
    // | gain | résolution | status |
    // | 1 | 12bits | 0 |
    // | 2 | 12 | 1 |
    // | 4 | 12 | 2 |
```

```
// | 8 | 12 | 3 |
// | 1 | 14bits | 4 |
// etc.
// | 1 | 16bits | 8 |
// etc.
// | 1 | 18bits | 12 |
// etc.
// -----
-----
// Conserver la valeur des trois derniers paramètres
uint8_t err = adc.convertAndRead(MCP342x::channel1,
MCP342x::oneShot,
MCP342x::resolution12,
MCP342x::gain4,
1000000, value, status);

if (err)
{
    Serial.print("Erreur de conversion : ");
    Serial.println(err);
}
else
{
    switch (status)
    {
    case 2:
        gain = 4;
        resolution = 12;
        break;
    case 6:
        gain = 4;
        resolution = 14;
        break;
    case 10:
        gain = 4;
        resolution = 16;
        break;
    case 14:
        gain = 4;
        resolution = 18;
        break;
    }
    Serial.print("value = ");
    Serial.println(value);
    Serial.print("Tension sur le canal 1 : ");
    tension_mV = value / (gain * pow(2, resolution - 12));
    Serial.print(tension_mV);
    Serial.println("mV");
    Serial.print("Temperature mesuree par le LM35 : ");
    Serial.print(tension_mV / 10.0);
    Serial.println("C");
}
}
```

```
delay(1000);  
}
```



Le projet pour l'IDE VSCode de l'exemple ci-dessus est téléchargeable [ici](#)

From:

<https://webge.fr/dokuwiki/> - **WEBGE Wikis**

Permanent link:

<https://webge.fr/dokuwiki/doku.php?id=materiels:can:mcp3424>

Last update: **2024/05/05 16:43**

