



## Specification document of MAX6605MXKV

Component manufacturer	Maxim Integrated
Model number	MAX6605MXKV
Datasheets	<a href="#">MAX6605 DS (maximintegrated.com)</a>
Specification Ver	01.00.00      Oct 04,2022      New release
Documentation provided	Rui Long Lab Inc. <a href="https://rui-long-lab.com/">https://rui-long-lab.com/</a>

1. Component datasheet .....	2
2. Component Software IF specification .....	3
3. File Structure and Definitions .....	5

### License

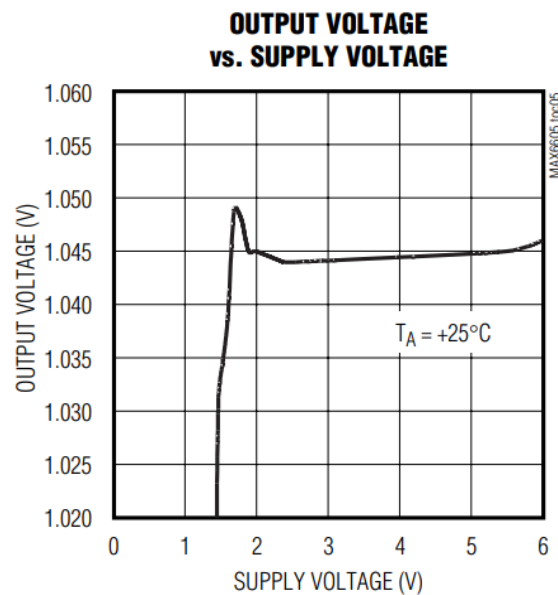
Open Source Software for Embedded Components ("OSS-EC") is open source software files and related documentation files for component products used in computer systems and other applications. OSS-EC is provided to those who accept the OSS-EC Terms of Use for the OSS-EC site; see [https://oss-ec.com/license\\_agreement/](https://oss-ec.com/license_agreement/) for the OSS-EC Terms of Use. By downloading the OSS-EC from the OSS-EC site or obtaining the OSS-EC by any means, you accept the Terms of Use. Please read and accept the Terms of Use before using the OSS-EC. If you do not agree to the Terms of Use, please do not use the OSS-EC. We reserve the right to change these Terms of Use at any time and for any reason. We strongly recommend that you review the Terms of Use periodically.

## 1. Component datasheet

Temperature accuracy	$\pm 0.75^{\circ} \text{ C } ( 25^{\circ} \text{ C } )$
Temperature range	$-40 \text{ to } +85^{\circ} \text{ C }$
Range of power supply voltage ( Vdd )	2.7 to 5.5[V]
Output voltage ( Vout )	Linear $11.9 \times \text{Vdd} / 3.3 \text{ [mV/}^{\circ} \text{ C] Typ.}$ Vdd = 3.3 [V] $0 [^{\circ} \text{ C}] \text{ } 0.744\text{[V] Typ.}$
Calculation	$\text{Vout} = 0.744\text{V} + ( 0.0119 \text{ V/}^{\circ} \text{ C} \times \text{Ta} )$ $\text{Ta} = ( \text{Vout} - 0.744\text{V} ) / 0.0119 \text{ V/}^{\circ} \text{ C}$

More accurate temperature calculation

$$\text{Vout} = 0.744\text{V} + ( 0.0119 \text{ V/}^{\circ} \text{ C} \times \text{Ta} ) + ( 1.604 \times 10^{-6} \times \text{Ta}^2 )$$



## Applications

IoT etc

- Cellular Phones
- Battery Packs
- GPS Equipment
- Digital Cameras

Automotive

## 2. Component Software IF specification

The software interface specifications based on the MAX6605MXKV component specifications are as follows.

The voltage value-to-physical value conversion equation is a linear conversion equation as shown in the equation below.

ADC value to voltage value conversion formula

$$v_i = ( a_i \times i_{ADC\_vdd} ) / 2^{i_{ADC\_bit}} \quad [V]$$

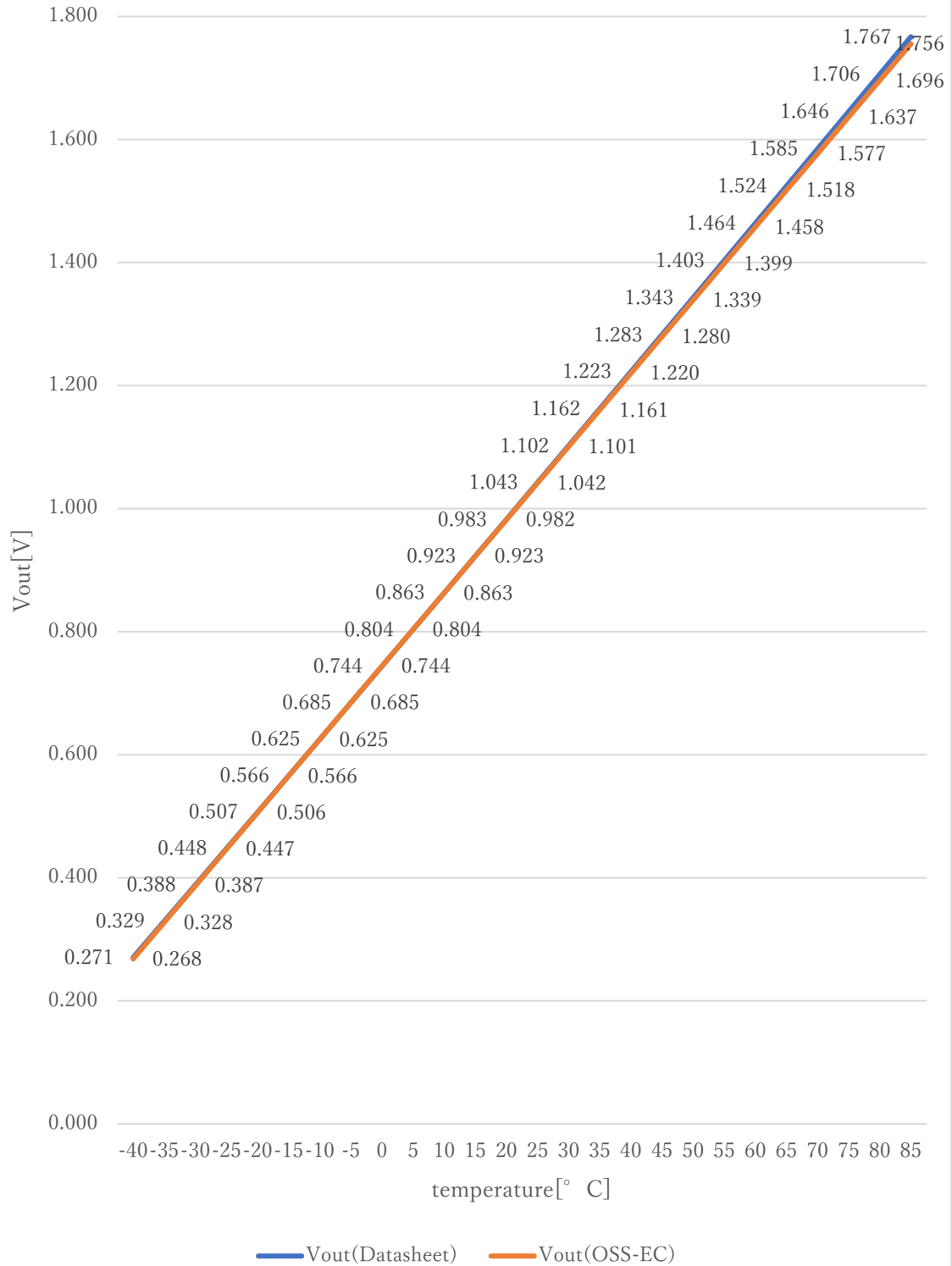
Voltage value to physical value conversion formula

$$y = ( v_i - i_{MAX6605MXKV\_xoff} ) / i_{MAX6605MXKV\_gain} + i_{MAX6605MXKV\_yoff} \quad [^{\circ}C]$$

$$i_{MAX6605MXKV\_min} \leq y \leq i_{MAX6605MXKV\_max}$$

$a_i$	A/D conversion value	
$v_i$	Sensor output voltage value [V]	
$i_{ADC\_vdd}$	Sensor supply voltage value [V]	
$i_{ADC\_bit}$	A/D conversion bit length	
$y$	Temperature value [ $^{\circ}C$ ]	
#define $i_{MAX6605MXKV\_xoff}$	<u>0.744F</u>	// X offset [V]
#define $i_{MAX6605MXKV\_yoff}$	<u>0.0F</u>	// Y offset [ $^{\circ}C$ ]
#define $i_{MAX6605MXKV\_gain}$	<u>0.0119F</u>	// Gain [V/ $^{\circ}C$ ]
#define $i_{MAX6605MXKV\_max}$	<u>85.0F</u>	// Temperature Max [ $^{\circ}C$ ]
#define $i_{MAX6605MXKV\_min}$	<u>-40.0F</u>	// Temperature Min [ $^{\circ}C$ ]

## Datasheet : OSS-EC



$$V_{out}(\text{Datasheet}) = 0.744\text{V} + (0.0119 \text{ V}/^{\circ} \text{C} \times T_a) + (1.604 \times 10^{-6} \times T_a^2)$$

### 3. File Structure and Definitions

#### MAX6605MXKV.h

```
#include "user_define.h"

// Components number
#define iMAX6605MXKV      111U           // Maxim Integrated MAX6605MXKV

// MAX6605MXKV System Parts definitions
#define iMAX6605MXKV_xoff  0.744F           // X offset [V]
#define iMAX6605MXKV_yoff  0.0F               // Y offset [°C]
#define iMAX6605MXKV_gain  0.0119F           // Gain [V/°C]
#define iMAX6605MXKV_max   85.0F             // Temperature Max [°C]
#define iMAX6605MXKV_min   -40.0F            // Temperature Min [°C]

extern const tbl_adc_t tbl_MAX6605MXKV;
```

## MAX6605MXKV.cpp

```
#include "MAX6605MXKV.h"

#if iMAX6605MXKV_ma == iSMA // Simple moving average filter
static float32 MAX6605MXKV_sma_buf[iMAX6605MXKV_SMA_num];
static const sma_f32_t MAX6605MXKV_Phy_SMA =
{
    iInitial , // Initial state
    iMAX6605MXKV_SMA_num , // Simple moving average number & buf size
    0U , // buffer position
    0.0F , // sum
    &MAX6605MXKV_sma_buf[0] // buffer
};

#elif iMAX6605MXKV_ma == iEMA // Exponential moving average filter
static const ema_f32_t MAX6605MXKV_Phy_EMA =
{
    iInitial , // Initial state
    0.0F , // Xn-1
    iMAX6605MXKV_EMA_K // Exponential smoothing factor
};

#elif iMAX6605MXKV_ma == iWMA // Weighted moving average filter
static float32 MAX6605MXKV_wma_buf[iMAX6605MXKV_WMA_num];
static const wma_f32_t MAX6605MXKV_Phy_WMA =
{
    iInitial , // Initial state
    iMAX6605MXKV_WMA_num , // Weighted moving average number & buf size
    0U , // buffer position
    iMAX6605MXKV_WMA_num * (iMAX6605MXKV_WMA_num + 1)/2 , // kn sum
    &MAX6605MXKV_wma_buf[0] // Xn buffer
};

#else // Non-moving average filter
#endif

#define iDummy_adr 0xffffffff // Dummy address

const tbl_adc_t tbl_MAX6605MXKV =
{
```

```

        iMAX6605MXKV
        iMAX6605MXKV_pin
        iMAX6605MXKV_xoff
        iMAX6605MXKV_yoff
        iMAX6605MXKV_gain
        iMAX6605MXKV_max
        iMAX6605MXKV_min
        iMAX6605MXKV_ma

#if iMAX6605MXKV_ma == iSMA // Simple moving average filter
    &MAX6605MXKV_Phy_SMA
    (ema_f32_t*) iDummy_adr
    (wma_f32_t*) iDummy_adr
#elif iMAX6605MXKV_ma == iEMA // Exponential moving average filter
    (sma_f32_t*) iDummy_adr
    &MAX6605MXKV_Phy_EMA
    (wma_f32_t*) iDummy_adr
#elif iMAX6605MXKV_ma == iWMA // Weighted moving average filter
    (sma_f32_t*) iDummy_adr
    (ema_f32_t*) iDummy_adr
    &MAX6605MXKV_Phy_WMA
#else // Non-moving average filter
    (sma_f32_t*) iDummy_adr
    (ema_f32_t*) iDummy_adr
    (wma_f32_t*) iDummy_adr
#endif

};

```