Getting Started With *ADA* Daughter Board

Using VHDL

Prof. Dr. Martin J. W. Schubert, Electronics Laboratory,

OTH Regensburg, Regensburg, Germany

Getting Started With *ADA* Daughter Board

Using VHDL

Abstract. This communication presents the *ADA* conversion board, which is presented as daughter board to DE1-SoC board, but is compatible with different *DEx* boards from *Terasic*, featuring the required 40 pin user header, e.g. *DE2, DE2-70, DE2-115*.

1 Introduction

1.1 Objectives and Organization of this Document

This document is intended for students learning time-discrete Signal processing, A/D and D/A conversion as well as electronic design automation using VHDL. It is a comprehensive example that teaches different aspects on the same system.

Naming conventions. Digital signal names containing strings *din* (digital in) or *dout* (digital out) respect the *ADA* board point of view. Signal *DAC3dout9* is a 9-level digital output signal from the *ADA* board and input to *DAC3*. Signal *adc_din* is output from the *ADC* and a digital input signal to the *ADA* board.

The organization of this document is as follows:

Section **1** is this **Introduction**,
Section **2** introduces the **A/D/A conversion board** used as example,
Section **3** presents the most important A/D and D/A conversion models,
Section **4** tests the DACs
Section **5** tests the flash ADC
Section **6** tests the flash ADC followed by DAC3 and measures quantization noise
Section **7** draws relevant **conclusions** and
Section **8** offers **references.**

1.2 Tools

1.2.1 *DE1-SoC* Hardware

This document assumes that you are familiar with the *Terasic's* [1] *DE1-SoC* board using an *Intel Cyclone V FPGA* [2] or a similar *DEx* board with the same general-purpose input/output (GPIO) user header. The version of your *DE1-SoC* board can be identified at [3]. *DE1-SoC* board revisions *F* and *G* differ in a printed company label only. To get it from the internet, go to [4] to find and download *DE1-SoC_v.5.1.2_HWrevF_SystemCD.zip* [5] and download it. It contains amongst other things important documents such as *DE1-SoC User Manual* [6] and *Schematic* [7]. On the computer system of OTH Regensburg you will also find the CD on drive K:\Sb\ [8]. Do not use any other manual revision to follow this documentation. The differences may be considerable.

1.2.2 *Quartus II* [9] and *ModelSim* [10] Software Tools

It is assumed that you have *Intel's Quartus II 13 [9]* and *ModelSim [10]* software available. To download this freeware for your private PC you have to sign in at *Intel* [11]. At OTH Regensburg's PC pools of faculties *EI* and *IM* this software is installed. At faculty *EI* also *Quartus II 8* is installed supporting sme older *DE2* boards with *Cyclone II* FPGAs, because they are not supported for *Quartus II* versions greater than 13.1.

1.2.3 Use of *VHDL*

The *IEEE standard VHDL Language reference manual* [12] is comprehensive and demanding to read. *Qualis VHDL Quick Reference Card* [13] and *1164 Packages Quick Reference Card* [14] are compact but difficult to understand. Feel free to find your own sources.

VHDL is not case sensitive. In the following, **`KEYWORDS`** will be written in **`ALL CAPITAL LETTERS`** and user defined names in **`lowercase letters`**. Exceptions are capitalized initials used for composed self-made names, e.g. *AddressBus* or *DataBus*. Self-made data types begin with *t_*, e.g. *t_StateVector*.

1.3 Acknowledgements

The author would like to thank *Terasic Technologies* [1] for admission to use screen copies of *Terasic* documentation for teaching purposes in this lectures.

`At 19.09.2014 08:49, Terasic - Dong Liu wrote:`

```
  Dear Martin,
  Thank you for using DE boards to teach VHDL. Yes, you can open all
  DE design resources for teaching purpose. Thank you!
  Best Regards,
  Doreen Liu
```

2 The *ADA* Daughter Board

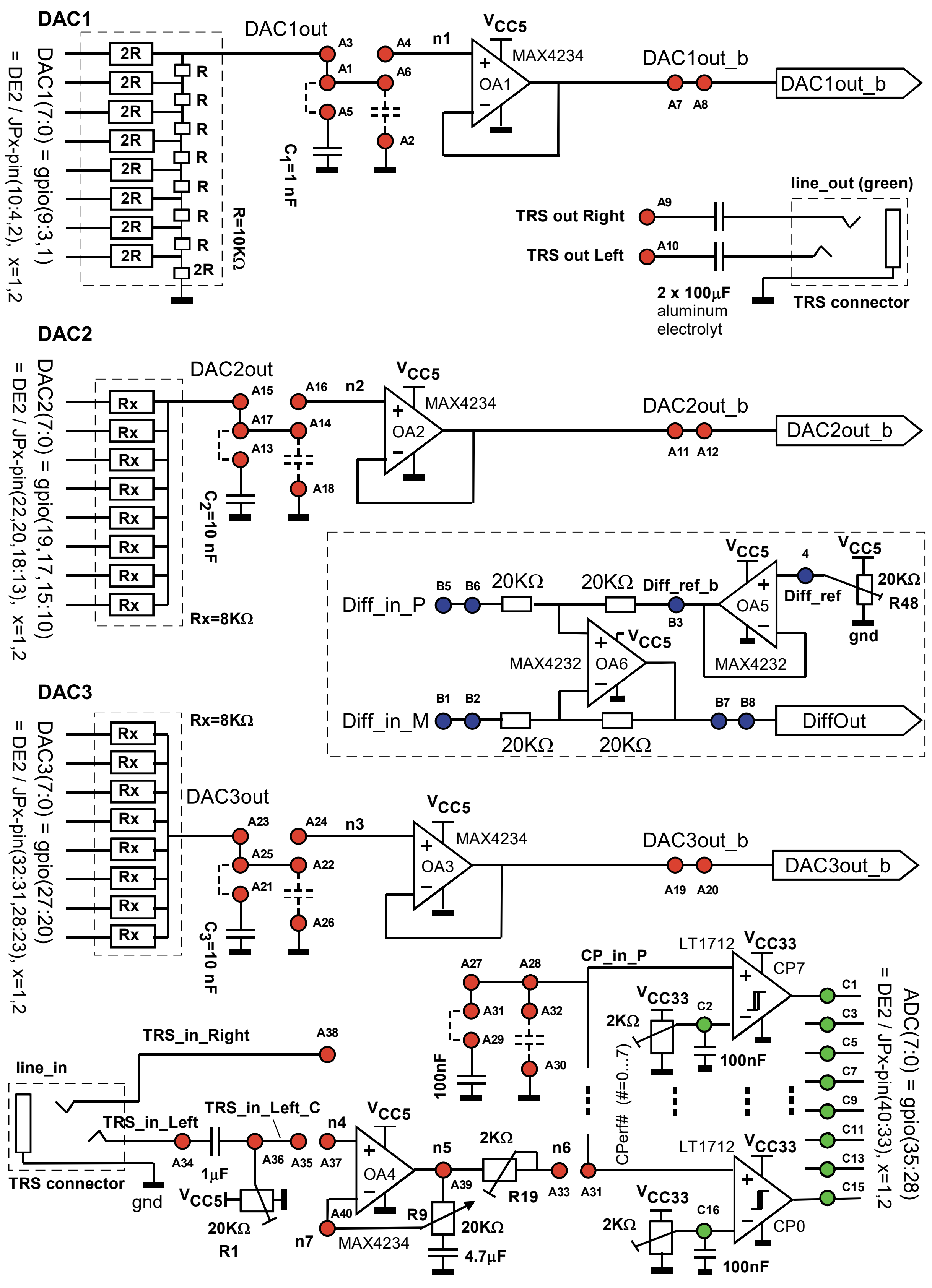


Fig. 2.1: Schematics of the *ADA* daughter board

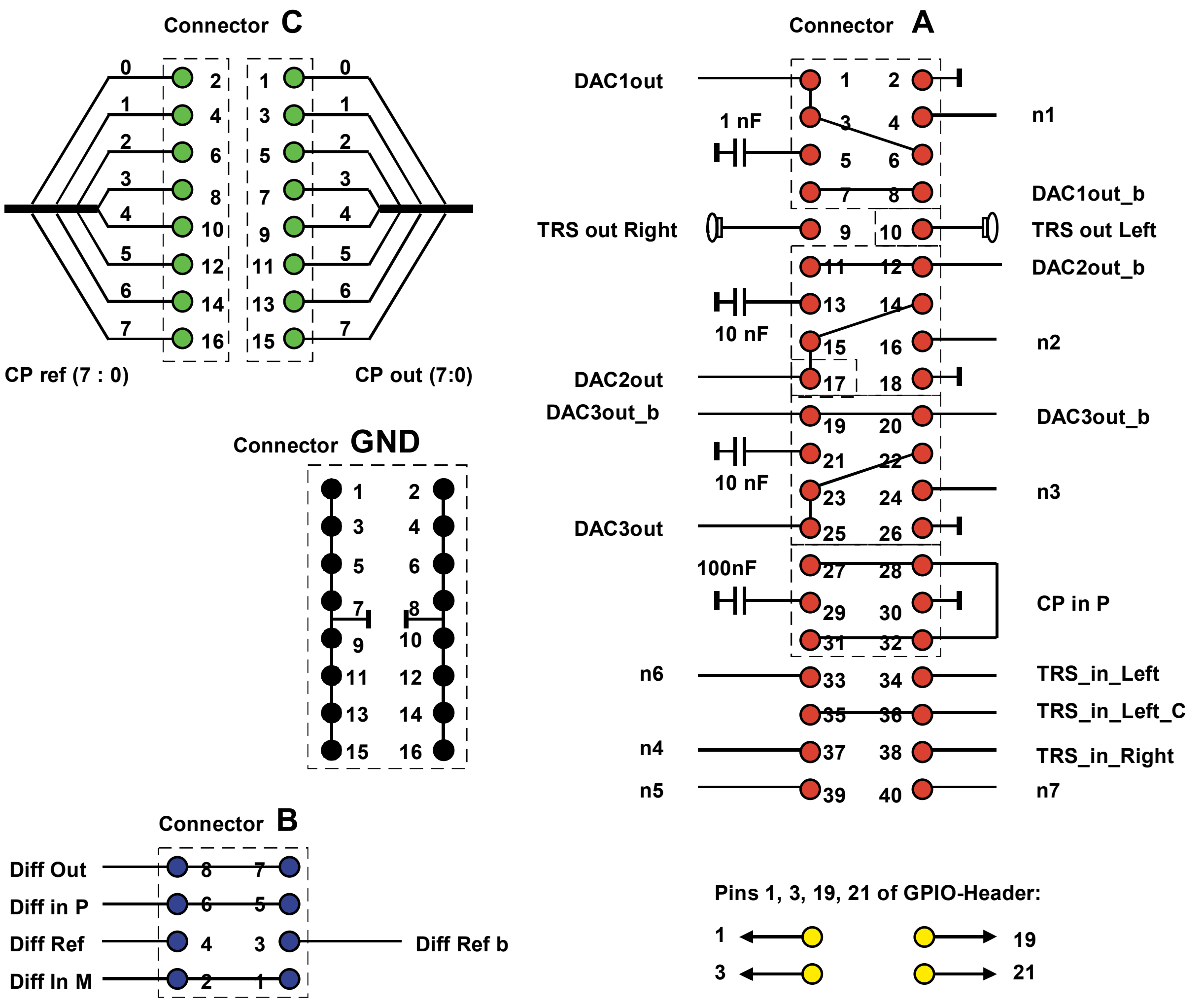


Fig. 2.2: Connectors of the *ADA* daughter board

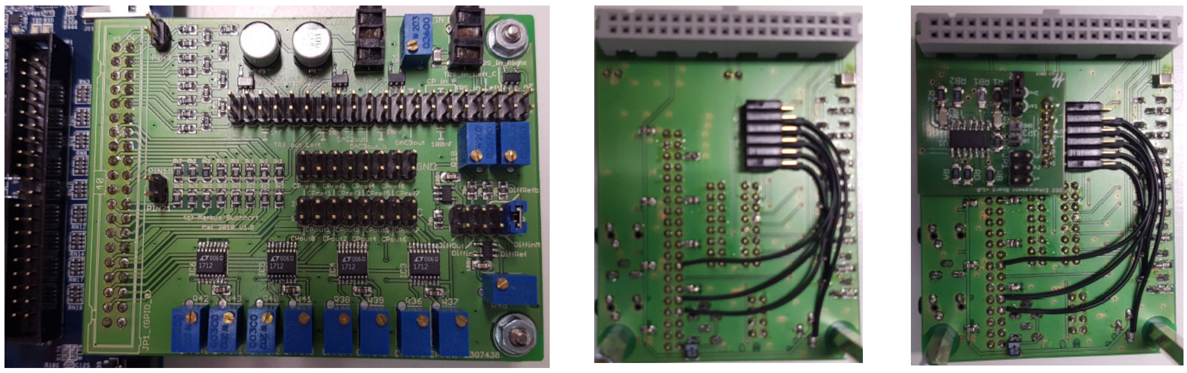


Fig. 2.3: ADA daughter board photos

- Fig. 2.1 shows the *ADA* daughter board schematics.
- Fig. 2.2 illustrates the *ADA* board from the user header connector point of view.
- Fig. 2.3 shows photos of the *ADA* board with and without *DSM* grandchild board.

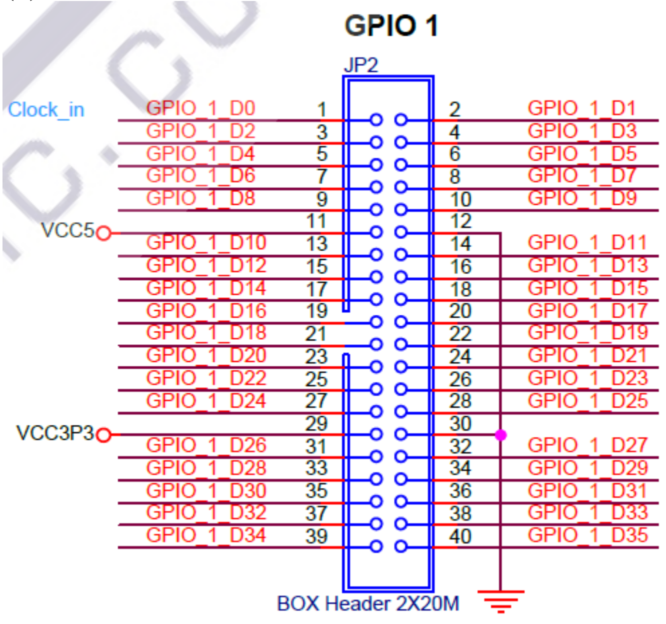
- Fig. 2.4 is related to the user headers of different *DEx* boards and maps the GPIO labels to the user header connector pins.
- Fig. 2.5 shows the production schematics of the *ADA* board with DAC schematics on the top part and other schematics on the bottom part of the figure.
- Fig. 2.6 the production layouts in top and bottom view, respectively.

Deactivate the *DSM* grandchild board under the *ADA* daughter board to avoid inferences when working with this document. This may be done in different ways, for example:

- Completely disconnect *DSM* grandchild board from *ADA* board → please be careful not to lose the small *DSM* board.
- Plug grandchild board into the unconnected parking position of the plug on the *ADA* board, which is the row with the bigger distance to the *ADA* board.

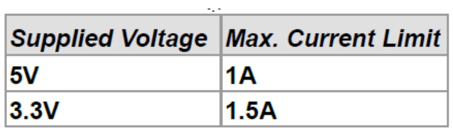
This tutorial is made for the *DE1-SoC* board. The user-header pins are assigned for compatibility to *DE2, DE2-70* and *DE-115* boards. Therefore pins 1, 3, 19, 21 of the user header remain unused. Figs. 2.4(a) copied from the *DE1-SoC* board's schematics [7] illustrates the user header's connectivity, while Fig. part (b) copied from the user manual [6], shows the current limitations of the on-board voltage sources V_{CC3P3} and V_{CC5}.

(a) User header of *DE1-SoC* board



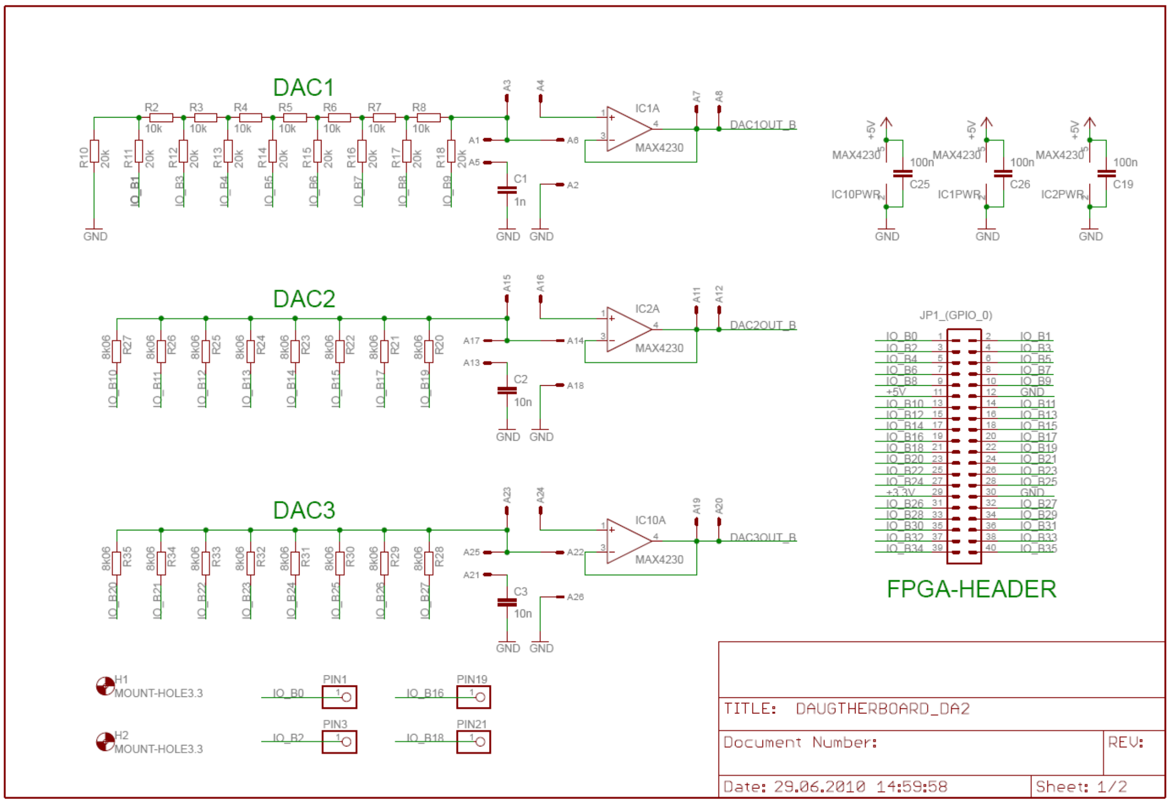
GPIO 1
JP2
Clock_in
VCC5
VCC3P3
GPIO_1_D0 1 2 GPIO_1_D1
GPIO_1_D2 3 4 GPIO_1_D3
GPIO_1_D4 5 6 GPIO_1_D5
GPIO_1_D6 7 8 GPIO_1_D7
GPIO_1_D8 9 10 GPIO_1_D9
11 12
GPIO_1_D10 13 14 GPIO_1_D11
GPIO_1_D12 15 16 GPIO_1_D13
GPIO_1_D14 17 18 GPIO_1_D15
GPIO_1_D16 19 20 GPIO_1_D17
GPIO_1_D18 21 22 GPIO_1_D19
GPIO_1_D20 23 24 GPIO_1_D21
GPIO_1_D22 25 26 GPIO_1_D23
GPIO_1_D24 27 28 GPIO_1_D25
29 30
GPIO_1_D26 31 32 GPIO_1_D27
GPIO_1_D28 33 34 GPIO_1_D29
GPIO_1_D30 35 36 GPIO_1_D31
GPIO_1_D32 37 38 GPIO_1_D33
GPIO_1_D34 39 40 GPIO_1_D35
BOX Header 2X20M

(b) Current limitations



Supplied Voltage	Max. Current Limit
5V	1A
3.3V	1.5A

Fig 2.4: *De1-SoC* user header *JP1*:
(a) Schematics copied from [7]
(b) Current limitations, copied from [6]



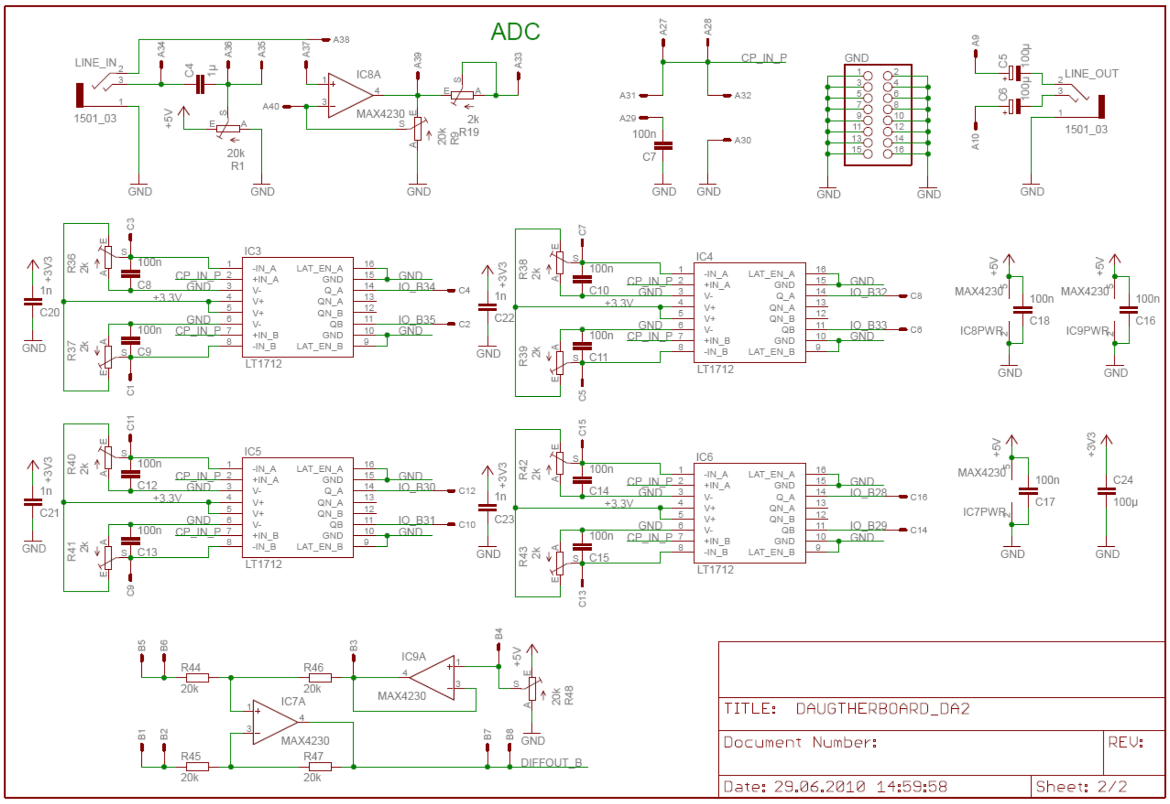


Fig. 2.5: *ADA* board schematics (by Markus Buchhart, 2010)

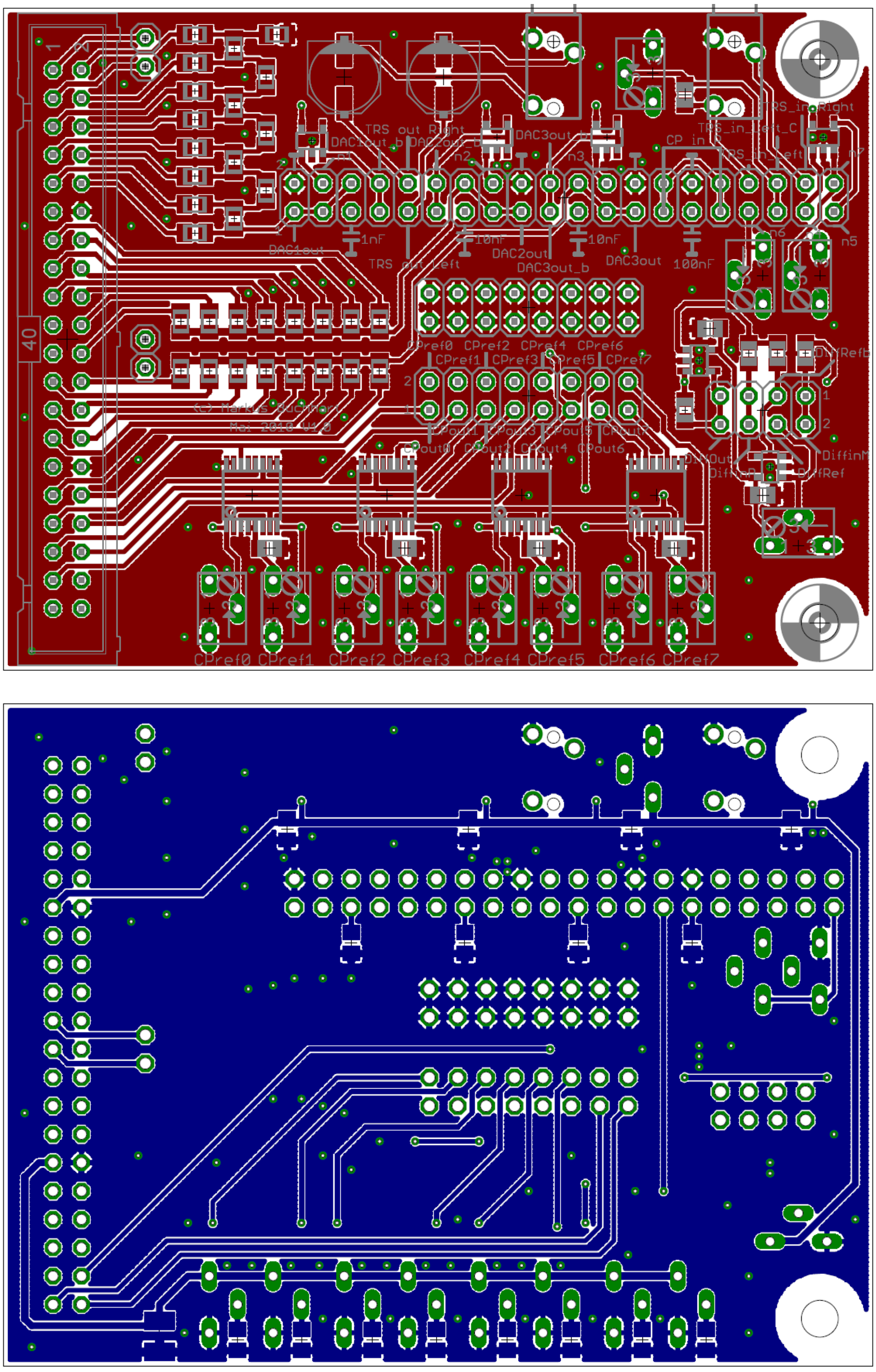


Fig. 2.6: *ADA* board **(red)** top and **(blue)** bottom layer layout (by M. Buchhart, 2010)

3 A/D and D/A Converter Modeling

Chapter 3.1 offer basic behavioral models of A/D and D/A conversion.
Chapter 3.2 models the flash DACs named *DAC2* and *DAC3* on the *ADA* daughter board.
Chapter 3.3 models the R2R DAC named *DAC1* used on the *ADA* daughter board

3.1 Basic Behavioral Models

We model a D/A converter (DAC) behaviorally as

$$U_{DAC,out} = \sum_{i=0}^{order} \Delta_i N_{DAC,in}^i$$

with $U_{DAC,out}$ and $N_{DAC,in}$ being output voltage and digital input word, respectively. According to signal processing linearity theorem, Δ_i=0 for $i \neq 1$, so that a linear DAC can be modeled as

$$U_{DACout} = \Delta_1 N_{in}$$

We model an A/D converter (ADC) behaviorally as

$$N_{ADC,out} = round\left(\sum_{i=0}^{order} \alpha_i U_{ADC,in}^i\right)$$

with $N_{ADC,out}$ and $U_{ADC,in}$ being digital output word and input voltage, respectively. According to signal processing linearity theorem, α_i=0 for $i \neq 1$, so that a linear ADC can be modeled as

$$N_{ADC,out} = round\left(\alpha_1 U_{ADC,in}\right).$$

3.2 Flash-DAC Theory

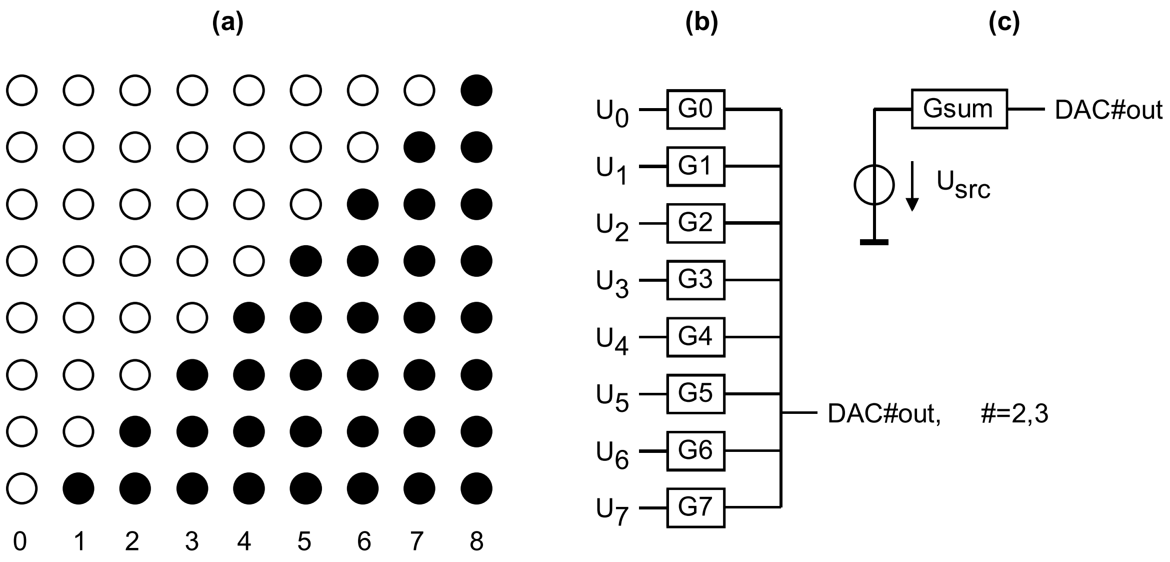


Figure 3.2.1: A 9-level thermometric code, **(b)** Flash DAC and **(c)** its equivalent model

In the figure above we have $Z_{out}^{-1} = G_{sum} = \sum_{j=0}^{NoL-2=7} G_j$ and $U_{src} = \sum_{j=0}^{NoL-2=7} \frac{G_j}{Gsum} U_j$.

DAC2 and *DAC3* in Fig. 3.2.1 *NoL*=9 being the number of levels from *NoL*-1=8-bit thermometric code to pins labeled *DAC2out* and *DAC3out*, respectively. What is their output resistance for ideal voltage sources and 8KΩ-resistors?

. . . .

Compute $U_{DAC\#out9}$ = f(*NoL*,V_{CC}) with *NoL* input voltages being at V_{CC} and the others at *GND*=0V? R_j=8KΩ for j=1...8. Label the 9 voltage levels in the figure below for $V_{CC}=V_{CC33}$=3.3V.

Figure 3.2.2:
Minimum DAC output step:

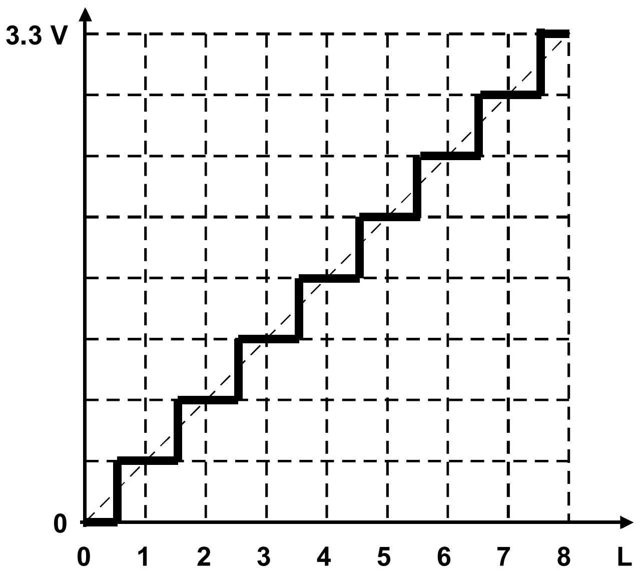
Δ_{DA} =

L=9 level DAC output voltage formula:

$U_{DAC\#out}$ =

with #=2,3. Amplification in V/bit:

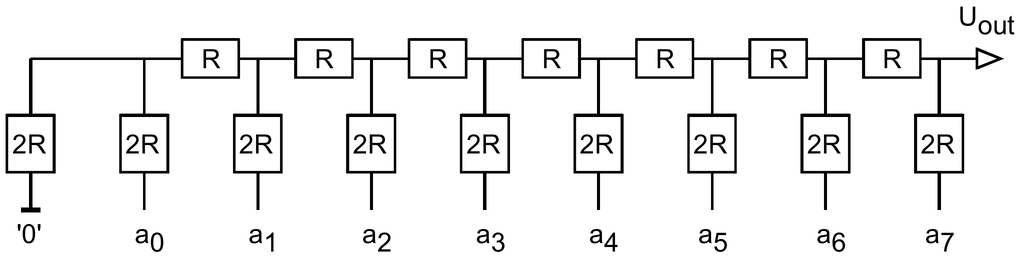
Δ_1 =



These *NoL*=9 level DAC with *NoL*-1=8 bits thermometric code seems to be not very efficient as 8 bits could be translated into 256 levels. However, combined with oversampling techniques such as dynamic element matching (DEM) and delta-sigma (ΔΣ) modulation this kind of DAC can represent an arbitrary number of levels with very high accuracy.

3.3 R2R-DAC Theory

Fig. 3.3:
R2R DAC



The R2R ladder DAC with output *DAC1out* delivers an output voltage of

$$U_{DAC1out} = \sum_{j=0}^{NoB-1=7} U_j \cdot 2^{j-8}$$

with *NoB* being the input bit-width and U_j either V_{CC3P3}=3.3V or 0V. As detailed in author's script on *ADA Converters* or *Schaltungstechnik* [15] the output impedance of the R2R-DAC in Fig. 3.3 is *R*.

The inaccuracy of the most significant bit (MSB) must be less than or equal to the half of the least significant bit (LSB). For 1% resistors this would be 6 bits, corresponding to an impact of $2^{-6}=1/64\cong1.56\%$. In other situations, e.g. processing acoustic signals, this holds true for the most significant used bit. Quiet passages of music may use only some LSBs.

4 Testing the Digital-to-Analog Converters (DACs)

Objective of this subchapter is to operate the DACs available on the ADA daughter board.

Look into the author's document *Getting Started with DE1-SoC Board* [x] or any other documentation to program the your *DEx* board (*x*=1,2…).

Hardware:

- Remove the *DSM* grandchild board connected below the DS board or remove all jumpers from the grandchild board DSM attached below daughter board ADA.
- Connect the *ADA* daughter board to the *gpio_1* expansion header of the *DEx* board. Thus, it will be controlled by signals *gpio*_1(35…0).

Software:

- Create a directory named *dexada_dac*. We will call it hereinafter *...\dexada_dac\*.
- Start the *Quartus II* software on your PC and create a project named dex*ada_dac* within it.
- Include a VHDL file named *dexada_dac.vhd* with the code from listing 4.1.

Some explanations to this code: A code line like

```
SIGNAL dac1dout256,dac2dout9,dac3dout9:std_logic_vector(7 DOWNTO 0);
```

is a non-executable declaration and must be located within the declaration region between VHDL keywords *IS* and *BEGIN*. Code lines

```
-- set drivers to the 3 DACs
dac1dout256 <= sw(7 DOWNTO 0);  -- input to 256-level DAC 1
dac2dout9   <= sw(7 DOWNTO 0);  -- input to   9-level DAC 2
dac3dout9   <= sw(7 DOWNTO 0);  -- input to   9-level DAC 3
```

are executable and must occur between VHDL keywords *BEGIN* and *END ARCHITCTURE.*

We avoid to use pins 1, 3, 19, 21 of the user headers for reasons of compatibility with the *DE2-70* board's user header, because this board assigns these pins to a PLL. Consequently, we avoid using signals

gpio_#(. . . . . . .) , *gpio_#*(. . . . . . .) , *gpio_#*(. . . . . . .), *gpio_#*(. . . . . . .), # = 0,1

signal *dac1dout256* controls the output voltage of the analog signal *DAC1out* on *ADA* board,
signal *dac2dout9* controls the output voltage of the analog signal *DAC2out* on *ADA* board,
signal *dac3dout9* controls the output voltage of the analog signal *DAC3out* on *ADA* board.

Check how the indices of the *gpio_1*(..) signal in listing 4.1 avoid these indices:

```
gpio_1( 9 DOWNTO  3) <= dac1dout256(     7 DOWNTO 1     );
gpio_1( 1)           <= dac1dout256(     0              );

-- drive DAC2:
gpio_1(19)           <= dac2dout9(     7              );
gpio_1(17)           <= dac2dout9(     6              );
gpio_1(15 DOWNTO 10) <= dac2dout9(     5 DOWNTO 0     );
```

observe in Fig. 2.4(a) how these general purpose input/output (gpio) signals are mapped to the pins of the user header.
Compile the code of listing 4.1 with *Quartus II* and download *dexada_dac.sof* into FPGA.

Listing 4.1: Testing the DACs

```
-- For Board: Altera Board DE1-SoC with FPGA Cyclone V 5CSEMA5F31C6N
LIBRARY ieee; USE ieee.std_logic_1164.ALL,ieee.std_logic_signed.ALL;
ENTITY dexada_dac IS
    PORT(CLOCK_50:IN std_logic;
      key:IN std_logic_vector(3 DOWNTO 0);      -- low when pressed
      sw:IN std_logic_vector(9 DOWNTO 0);       -- low when pulled down
      ledr:BUFFER std_logic_vector(9 DOWNTO 0); -- high active
      hex0,hex1,hex2,hex3,hex4,hex5:OUT std_logic_vector(0 TO 6);
      gpio_0:BUFFER std_logic_vector(35 DOWNTO 0);
      gpio_1:INOUT std_logic_vector(35 DOWNTO 0)
    );
END ENTITY dexada_dac;

ARCHITECTURE rtl_dexada_dac OF dexada_dac IS
  TYPE t_7seg IS ARRAY(0 TO 15) OF std_logic_vector(0 TO 6);
  CONSTANT c7seg:t_7seg:=("1111110", "0110000", "1101101", "1111001",
    "0110011", "1011011", "1011111", "1110000", "1111111", "1110011",
    "1110111", "0011111", "1001110", "0111101", "1001111", "1000111");
  SIGNAL dac1dout256,dac2dout9,dac3dout9:std_logic_vector(7 DOWNTO 0);
BEGIN
  --
  -- read switches to set drivers to the 3 DACs
  dac1dout256 <= sw(7 DOWNTO 0);  -- input to 256-level DAC 1
  dac2dout9   <= sw(7 DOWNTO 0);  -- input to   9-level DAC 2
  dac3dout9   <= sw(7 DOWNTO 0);  -- input to   9-level DAC 3
  --
  -- drive DAC1:
  gpio_1( 9 DOWNTO  3) <= dac1dout256(     7 DOWNTO 1     );
  gpio_1( 1)           <= dac1dout256(     0              );
  --
  -- drive DAC2:
  gpio_1(19)           <= dac2dout9(     7               );
  gpio_1(17)           <= dac2dout9(     6               );
  gpio_1(15 DOWNTO 10) <= dac2dout9(     5 DOWNTO 0      );
  --
  -- drive DAC3:
  gpio_1(     27 DOWNTO 20     ) <= dac3dout9;
  --
  -- copy to gpio_0 available for monitroing purposes
  gpio_0 <= gpio_1;
  --
  -- control LEDs
  ledr <= sw;  -- same length -> no indexing required
  --
  -- Control HEX-displays
  p_check_hex:PROCESS(sw(0))
  BEGIN
    IF sw(0)='0' THEN
      hex0<=NOT c7seg(0); hex1<=NOT c7seg(1); hex2<=NOT c7seg(2);
      hex3<=NOT c7seg(3); hex4<=NOT c7seg(4); hex5<=NOT c7seg(5);
    ELSE
      hex0<=NOT c7seg(6); hex1<=NOT c7seg(7); hex2<=NOT c7seg(8);
      hex3<=NOT c7seg(9); hex4<=NOT c7seg(10); hex5<=NOT c7seg(11);
    END IF;
  END PROCESS p_check_hex;
END ARCHITECTURE rtl_dexada_dac;
```

4.1 Testing *DAC1*

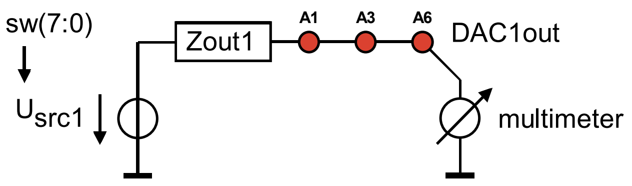


Fig. 4.1: Test setup for *DAC1*.

4.1.1 Measuring the Equivalent Inner Source Voltage U_{src1}

Connect a voltmeter to output of *DAC1*, which is drawn as source voltage and output impedance in Fig. 4.1. As the voltmeter has a high input impedance, we measure *DAC1's* equivalent source voltage U_{src1}. Use switches *sw0 ... sw7* for the following questions:

V_{DD} theoretically: 3.3V, (available e.g. at pin 29 of *GPIO* 1,2) , measured . . . .

U_{src1} for *DAC1* varies in . . . . steps from a minimum voltage of V

to a theoretical max. voltage of . . . . , measured . . . .

The resolution is theoretically: . . . ., , measured . . . .

Look into the data sheet of the max423x operational amplifier [16]:

Common mode input voltage range: . . . .

Typical output voltage swing at R_L=200Ω: . . . .

Input bias current: . . . .

Input offset voltage: typical: . . . . max: . . . .

On *ADA* board: Set jumper *A3-A4*. Does buffer *OA1* drive *DAC1out*_b? . . . .

Input offset voltage of amplifier *OA1* (measure e.g. between pins *A1–A7*)? . . . .

4.1.2 Measuring the Equivalent Output Impedance Z_{out1}

Expected output impedance from Fig. 3.3: $Z_{out1,ideal}$ = ...

Method A:
Set any voltage U_{src1}>0V (e.g. V_{DD}/2=1.65V), measure it: U_{src1} = ...

Switch from voltmeter to ampere-meter and measure the output current: I_{out1} = ...

The output impedance is $Z_{out1} = U_{src1} / I_{out1}$ = ...

Method B:
Set voltage U_{src1}=0V and measure with an Ohm-meter versus ground: Z_{out1} = ...

4.2 Testing *DAC2*

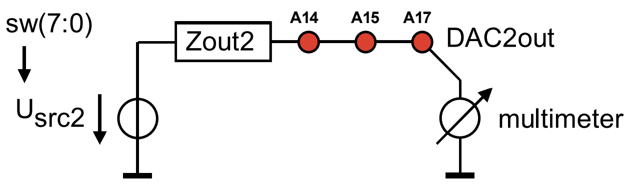


Fig. 4.2: Test setup for *DAC2*.

OA2 o.k.?. . . ., Offset voltage: . . . . . .

4.2.1 Measuring the Equivalent Inner Source Voltage U_{src2}

Use switches *sw0* ... *sw7* for the following questions:

U_{src2} for *DAC2* varies in . . . . steps from a minimum voltage of V

to a theoretical max. voltage of . . . . , measured . . . .

The resolution is theoretically: . . . . , measured . . . .

4.2.2 Measuring the Equivalent Output Impedance Z_{out2}

Expected output impedance from Fig. 3.2.1(b): $Z_{out2,ideal}$ = ...

Method A:
Set any voltage U_{src1}>0V (e.g. $V_{DD}/2$=1.65V), measure it: U_{src2} = ...

Switch from voltmeter to ampere-meter and measure the output current: I_{out2} = ...

The output impedance is $Z_{out2} = U_{src2} / I_{out2}$ = ...

Method B:
Set voltage U_{src2}=0V and measure with an Ohm-meter versus ground: Z_{out2} = ...

4.3 Testing DAC3

OA3 o.k.? . . . . . , Offset voltage: . . . .

4.3.1 Measuring the Equivalent Inner Source Voltage U_{src3}

Set *DSM* grandchild board under this *ADA* board into the unconnected parking position or remove it from *ADA* (→ see Fig. 2.3). Use switches *sw0* ... *sw7* for the following questions:

U_{src3} for *DAC3* varies in . . . . steps from a minimum voltage of V

to a theoretical max. voltage of . . . . , measured . . . .

The resolution is theoretically: . . . . , measured . . . .

4.3.2 Measuring the Equivalent Output Impedance Zout3

Expected output impedance from Fig. 3.2.1(b): $Z_{out3,ideal}$ = ...

The measured output impedance is Z_{out3} = ...

Clean up the working directory: delete all files not ending **.vhd, *.qpf, *.psf, *.cfd [, *.sof]*

5 Testing the Analog-to-Digital Converter (ADC)

5.1 Software: Read ADC Output and Visualize it with LEDs

Create a working directory named …/*dexada_adc*/. Create with a *Quartus II* project with the same name in it. There are two possibilities:

(i) Create a new project, use the VHDL code of listing 4.1, change all strings "*dexada_dac*" to "*dexada_adc*" and save it as *dexada_adc*.vhd.

(ii) Copy directory .../*dexada_dac*/ to .../*dexada_adc*/. Remove all files except the 4 with extension **.vhd, *.qpf, *.qsf, *.vhd*. Change all strings "*dexada_dac*" to "*dexada_adc*" in all file names an in all ASCII codes of the files.

Check if you can compile the project after the above renaming. You should get same results.

Read the comparator's digital output

Insert in the VHDL architecture's declaration region (somewhere between *IS* and *BEGIN*) in code line

```
SIGNAL adc_din:std_logic_vector(7 DOWNTO 0);
```

and include after keyword *BEGIN* the code line reading the ADC's output to signal *adc_din*:

```
--
-- reading ADC output
adc_din <= gpio_1(                          );
--                  .........................
```

To make the ADC's digital output-bits visible with LEDs 7…0 replace line

~~ledr <= sw; -- same lengths -> no indexing required~~

by lines

```
ledr(9 DOWNTO 8) <= sw(9 DOWNTO 8);
ledr(7 DOWNTO 0) <= adc_din;
```

Set jumper A25-A27 as shown in Fig. 5.1 to drive the ADC's input with the output of DAC3.

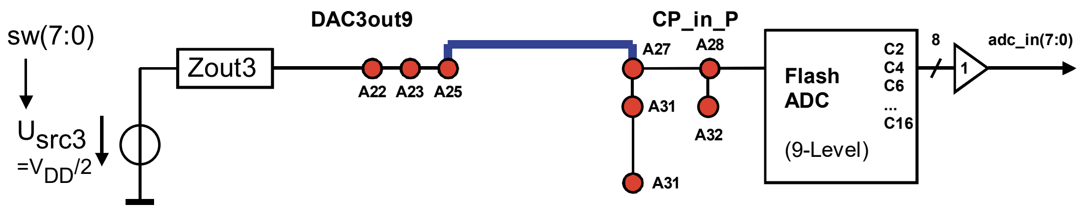


Figure 5.1: Using *DAC3out* as Voltage Source

5.2 Hardware: Trimming the Flash-ADC

The 8 threshold voltages are labeled on the *ADA* board with *CPref0,...,CPref7* (test points *C2*, *C4*, ... *C16*). Compute the 8 thresholds of Fig. 5.2 and tune the 8 potentiometers (potis) to the computed values. *CPref0* is the lowest and *Cpref7* the highest threshold voltage.

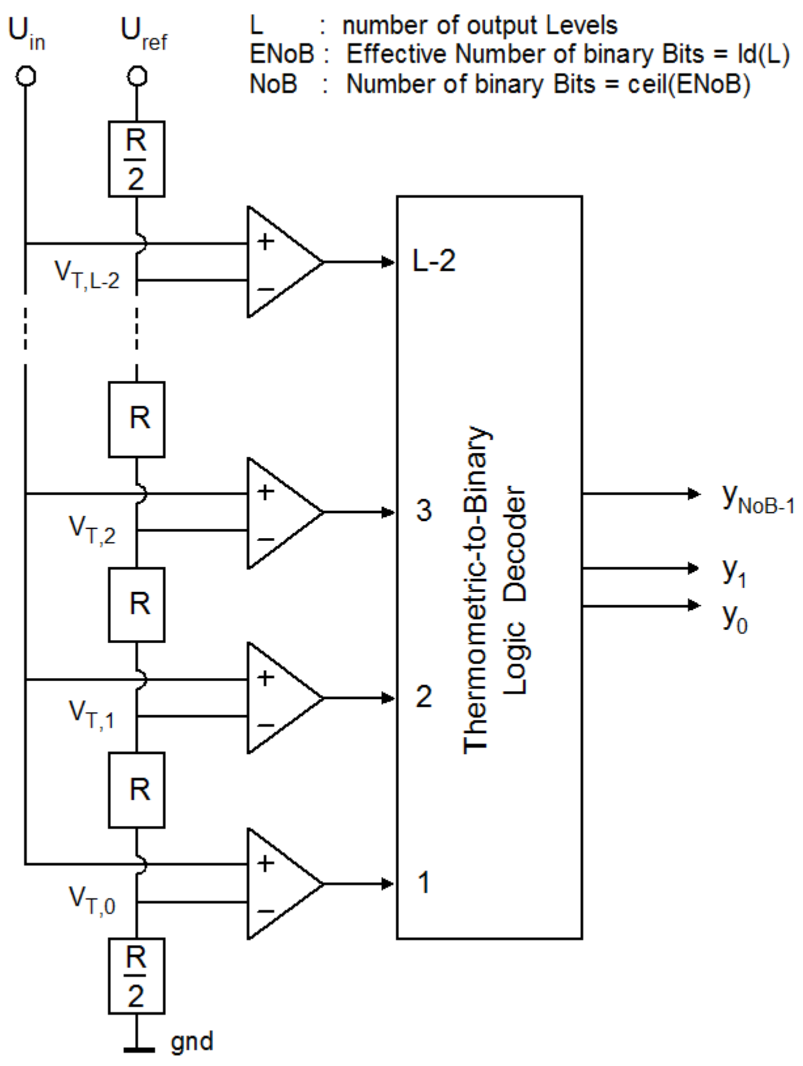
The 8 positive comparator inputs the are connected and labeled *CP_in_P* having several test points. Ideally, the Flash-ADC would have an infinite impedance and zero input current. However, the comparators have input bias currents.

Look into the LT1712 data sheet [17]. What is its typical input bias current?

. . . .

What is the typical input bias current of the total Flash-ADC?

. . . .



U_{in}
U_{ref}
L : number of output Levels
ENoB : Effective Number of binary Bits = ld(L)
NoB : Number of binary Bits = ceil(ENoB)
R/2
R
$V_{T,L-2}$
$V_{T,2}$
$V_{T,1}$
$V_{T,0}$
L-2
3
2
1
Thermometric-to-Binary Logic Decoder
y_{NoB-1}
y_1
y_0
gnd

Fig. 5.2: Basic principle of a flash ADC. The *m:n* decoder logic can be realized as simple summation of the comparator's output bits.

Write formula $V_{T,i}$=f(i=0...7,V_{CC}) for the 8 required threshold voltages of the comparators delivering the *NoL*-1=8 decision in Fig. 5.3. Write the $V_{T,i}$ to the right side of Fig. 5.3.

Figure 5.3:
NoL=9 level DAC output voltage formula:

$U_{DAC\#out} = i{\cdot}\Delta_{DA}$, i=0...L-1

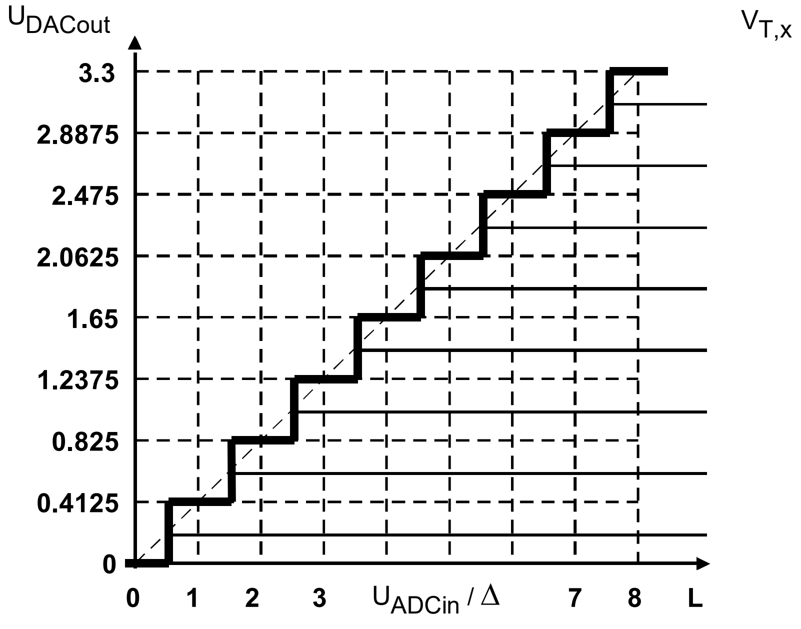
Find: L=9 level DAC input threshold formula using $U_{ref}=V_{CC}$=3.3V, i=0...L-2.

$\boldsymbol{V_{T,i}}$ =

with Δ_{AD} =

Amplification α_1 in bit/V:

α_1 =



- Remove DSM grandchild board under ADA board plug it into the unconnected position. The DSM grandchild board must not drive voltages into these measurements.
- Set $U_{DAC3out}$ to $V_{DD}/2$, using for example *sw*(7…0)="00001111", and measure the voltage at test point *A25*. Write U(*DAC3out*) in box "unconnected" in table 5.2.
- Connect $U_{DAC3out}$ to the Flash-ADC's input *CP_in_P* (that must not have other connections) by shorting pins *A25* and *A27*. Write $U_{DAC3out}$ in box "*DAC3out=CP_in_P* " of table 5.2.

Table 5.2: Impact of comparator's input bias currents on output impedance of *DAC3*.

DAC3out:	unconnected	*DAC3out=CP_in_P*	Difference
U(*DAC3out*):			

How do you explain the voltage jump when pins *A25* and *A27* connected?

. . . .

. . . .

Control $U_{DAC3out}$ with switches *sw*(7:0), measure it with a voltmeter and observe diodes *ledr*(7:0). What correlation do you observe between the number of switches *sw*(7:0) driving logical '1' and the number of LEDs *ledr*(7:0) turned on?

. . . .

6 Characterization Using an A/D/A Conversion System

Goal of this subchapter is to do some characterization like computation of quantization noise by assembly of an A/D/A conversion system according to Fig. 6.1.

6.1 Hardware Setup

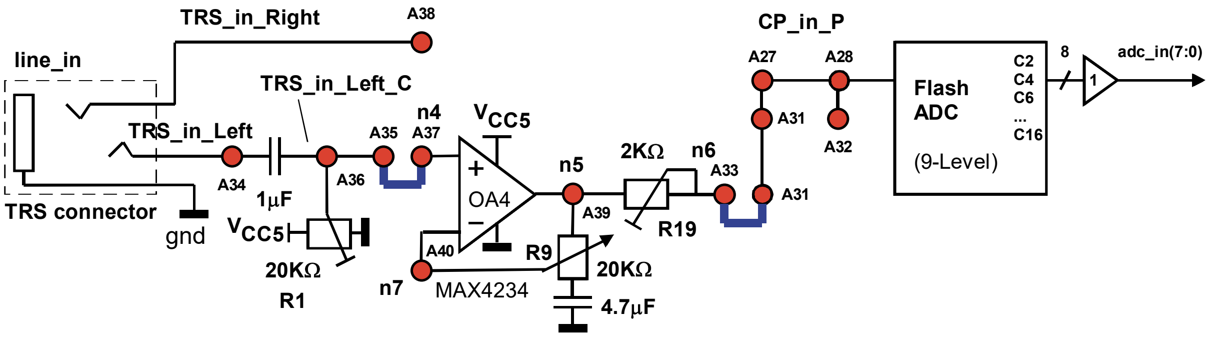


Fig. 6.1: Signal path: feed 250mV to *TRS_in_Left* (A34), amplify x 10 to *CP_in_P*.

In the previous chapter, we have driven the Flash-ADC with DAC3. Now we drive DAC3 with the Flash-ADC.

Feed a 1000 Hz sinusoidal signal with some 290mV peak-peak to node *TRS_in_Left* (A34). Observe on node *n5* (A39) a 10 times bigger amplitude. Use poti *R1* for DC balancing and poti *R9* for amplitude adjustment.

- In the Electronics Laboratory of OTH Regensburg, use Oscilloscope DSO-X2024 as waveform generator: Press button ***Wave Gen***, then use buttons below the screen to select your settings.
- If you have no waveform generator available, you may for example use your soundcard.

6.2 Software

A. Create Project *dexada_ada* and Program *DE1-SoC* board with VHDL Source Code:
Create directory *de1soc_ada* and project *de1soc_ada* from directory and project *dexada_adc*. To output the information of the 9-level Flash-DAC immediately via *DAC2* and *DAC3*, replace within file *de1soc_ada.vhd* code lines

```
~~  dac2dout9 <= sw(7 DOWNTO 0); -- input to 9-level DAC 2~~
~~  dac3dout9 <= sw(7 DOWNTO 0); -- input to 9-level DAC 3~~
```
by
```
  dac2dout9  <= adc_din;        -- output ADC data to DAC2
  dac3dout9  <= adc_din;        -- output ADC data to DAC3
```

compile it with *Quartus II* and download it into the *Cyclone II* FPGA on the *DE2* board. Connect *CH2* of your oscilloscope with *DAC3out* (e.g. test point *A25*). This voltage curve corresponds to the green staircase line in Fig. 6.3(b).

B. Circuit Assembly according to Fig. 6.3(a) on the *ADA* daughter board:

Hint: TRS = Tipp-Ring-Sleeve connectors are the ones typically used for speakers.

1. Remove jumper *A25–A27* to disconnect *DAC3out* from *CP_in_P*.
2. Use Ω-meter (between test points *A33-A39*) to adjust *R19* to ca. 100 Ω.
3. Shorten (jumper) test points *A31-A33* to drive *CP_in_P* with OpAmp *OA4* through *R19*.
4. Shorten (jumper) test points *A35-A37* to drive OpAmp *OA4*'s IN^+ with the soundcard.
5. Use poti *R1* to set *CP_in_P* to a DC bias voltage of $V_{DD}/2$=1,65V.
6. Drive a sinusoidal signal of ca. 250 mV_{pp} to *TRS_in_left* (*A34*).
7. Connect your oscilloscope's CH1 to *CP_in_P* (*A27*) and CH4 to *TRS_in_Left_C* (*A36*).
8. Use poti *R9* to adjust the amplification of *OA4* to a factor 10 or 20dB. (See → D.)

C. Test Tone Generation (Prefer hardware to generate test tones when available.)

If no hardware test tone generator is available you may use your sound card. Software is free available in the internet, e.g. *Test Tone Generator* [10], [11].

(a) After a quick installation you will see the window shown in Fig. 6.3(b).

(b) Set the tone duration to 9999s, a sinusoidal 1000Hz tone and some –21dBFS to get a peak-to-peak output amplitude of ca 250 mV_{pp}. Click the *ON* button.

(c) Connect your PC's soundcard output (green) with the *ADA* boards TRS *line-in* jack.

D. With the Oscilloscope:

I. Oscilloscope: CH1 has 500mV/dev and CH4 has 50mV/dev.

II. According to point B.7 above CH1 has to show the 10 x amplified signal of CH4.

III. Use poti R9 to adjust the curves at CH1 and CH4 to the same amplitude.

IV. U(*CP_in_P*) shown at oscillator channel CH1 corresponds to the yellow sinusoidal curve of the screen-shot in Fig. 6.3(b).

V. Get the sinusoidal (yellow) and staircase (blue) curves shown in Fig. 6.3(b) on your oscilloscope. Adjust generator's amplitude and DC-balance poti R1 to get Fig. 6.3(b).

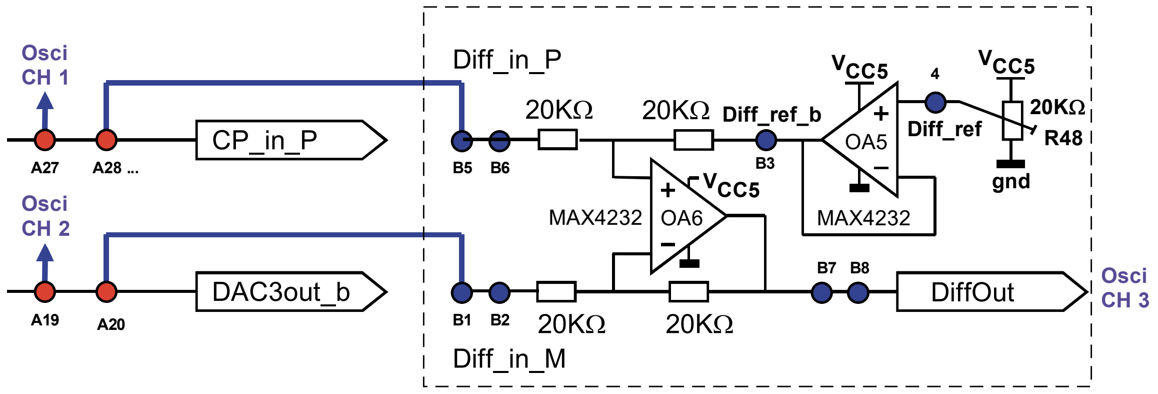
If the sinusoidal tone works well, drive on ADA board *DAC3out* and additionally *DAC2out* via the buffer amplifier *OA2* to *DAC2out_b* and connect this to *TRS out Left*. Plug a speaker to the *ADA* board's TRS *line_out* jack. Do you hear the test tone? . . . .

Disconnect the *Test Tone Generator* and play some music. Sound quality? . . . .

8 levels correspond to 3 bits. Our 9 levels correspond to ld(9) = ln(9)/ln(2) = ... bits.

6.3 Measuring the Quantization Noise

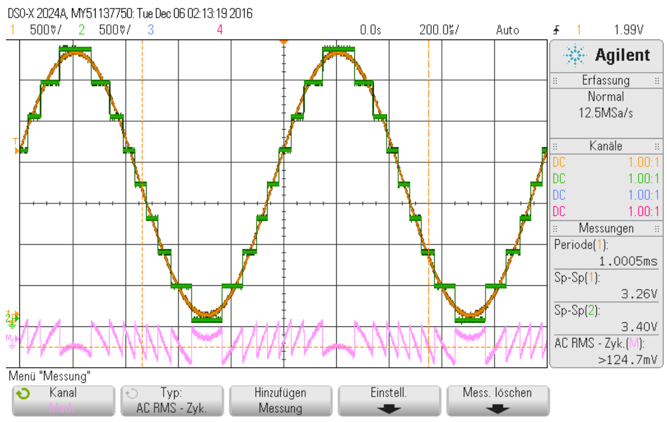
Goal of this subchapter is to make the difference (= quantization noise) between input and output curve visible as shown in the bottom (pink) curve of Fig. 6.3(b).



(a) Top: Circuit assembled on ADA board.

(b) Right: Oscilloscope screen shot: (i) yellow, sinusoidal: *CP_in_P*, (ii) blue in 9 levels: *DAC3out*, *(iii)* green, bottom: Difference =quantization noise: U(*B7*,*B3*).

Figure 6.3: Measuring quantization noise


DSO-X 2024A, MY51137750: Tue Dec 06 02:13:19 2016

Erfassung: Normal 12.5MSa/s

Messungen:
Periode(1): 1.0005ms
Sp-Sp(1): 3.26V
Sp-Sp(2): 3.40V
AC RMS - Zyk.(M): >124.7mV

We now want to measure the effective quantization-noise voltage, which is the red curve in Fig. 6.3(b), computed as difference between the green and the yellow curve.
We should have the sinusoidal (yellow) and staircase (green) curves of Fig. 6.3(b) on the scope.

Possibility A: Measure with the Oscilloscope DSO-X 2024A
Compute the Quantization noise by the oscilloscope's math menu:

- Compute difference: Math → Function f(t), Operator → subtract → CH1 – CH2
- Measure computed rms voltage: Meas → Type → AC-RMS Zyk

Possibility B: Using the hardware shown in Fig. 2.1(a)

- Adjust poti *R48* such that U(*Diff_ref_b*)=V_{DD}/2=1.65V.
- Connect *CP_in_P* to *Diff_in_P* and *DAC3out_b* to *Diff_in_M*.
- Display U(*DiffOut*) on oscilloscope as shown with the bottom (pink) curve in Fig. 6.3(b).
  Note: To get the 1.65V DC biasing voltage out of the quantization noise measurement you have to either use the AC oscilloscope channel input mode(see figure) or to measure between points *DiffOut* and *Diff_ref_b*.

Mathematically: $\frac{\Delta}{\sqrt{12}} = \frac{3.3V/8}{\sqrt{12}}$ **=** Measured:

. . . .

7 Conclusions

This communication introduces into the *ADA* board, offering 1 flash ADC and 3 DACs. The *ADA* board is operated as daughter board of *Terasic's DE1-SoC* board, programmed with *Quartus II Rev. 18* [9]

8 References

[1] Available: https://www.terasic.com.tw.
[2] Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=836
[3] Available: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=886
[4] Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=165&No=836&PartNo=4
[5] CD ROM *DE1-SoC CD-ROM (rev.F Board) Version 5.1.2 of 2910-01-28* from [4]
[6] *DE1*-SoC User Manual, Ref. F, taken from [5]
[7] *DE1*-SoC Schematic, Ref. F, taken from [5]
[8] K:\SB\Sources\EDA\Terasic\Hardware\
[9] Available: https://en.wikipedia.org/wiki/Intel_Quartus_Prime
[10] Available: https://en.wikipedia.org/wiki/ModelSim
[11] Av. https://www.intel.com/content/www/us/en/programmable/downloads/download-center.html
[12] Available: https://edg.uchicago.edu/~tang/VHDLref.pdf
[13] Available: https://www.mimuw.edu.pl/~marpe/pul/card_vhdl.pdf
[14] Available: https://www.mimuw.edu.pl/~marpe/pul/card_1164.pdf
[15] Available: https://hps.hs-regensburg.de/scm39115/
[16] MAX4230 - MAX4324 data sheet, *Maxim Integrated*, Available: https://datasheets.maximintegrated.com/en/ds/MAX4230-MAX4234.pdf
[17] LT1711 / 1712 Rail-to-Rail Comparators, Analog Devices (Linear Technlogy) Available: https://www.analog.com/media/en/technical-documentation/data-sheets/171112f.pdf