ANALOG I/O BOARD FOR PC'S

Jos Groot (InterNet: jos.groot@fel.tno.nl)
September 24, 1992

This note describes how to build a low cost extension board containing an analog-to-digital (ADC) and a digital-to-analog (DAC) converter, for use in a standard PC XT extension slot. The board utilizes the 8-bit AD7569 IC, containing a 1 μs DAC, a 2 μs ADC and a track/hold circuit with 200 kHz bandwidth. The part costs amount to 80-100 US$, approximately. The board can for example be used to sample sound with the ADC and play it back through the DAC, possibly after some 'on-the-fly' digital signal processing. Example Turbo Pascal routines for programming the board are included in this description, of which two implement electric guitar sound effects.

1 Operation

This part explains how the card operates (see accompanying electronic diagram contained in the ADDA10.ZIP archive).

1.1 Buffering

The card is put in a standard 62-contact XT extension slot. The slot is drawn in the upper left of the diagram.

The data lines SD0-SD7 are buffered by a bidirectional 3-state 74LS245. The buffer direction is determined by the DIR input connected to $\overline{\mathrm{IOR}}$ of the slot. It is from B to A during an I/O read ($\overline{\mathrm{IOR}}$=DIR low), and from A to B during an I/O write ($\overline{\mathrm{IOR}}$=DIR low). Both ports are in a high-impedance state (virtually disconnected from the circuit) when their is no access to the card.

The control lines $\overline{\mathrm{IOR}}$, $\overline{\mathrm{IOW}}$, RESET and AEN are buffered by the unidirectional 3-state buffer 74LS244. It is always enabled because $\overline{\mathrm{1G}}$ en $\overline{\mathrm{2G}}$ are low.

The address lines SA2-SA9 are unbuffered because these are only loaded by the 8-bit 74LS688 comparator. The unbuffered address lines SA0 en SA1 are loaded by 2 Schottky TTL loads each, while a maximum of 1 is recommended. This will cause no problems in general, certainly not when not all extension slots are used.

1.2 Address decoder

The address decoder section consists of an 8-bit comparator 74LS688, a 74LS02 containing 4 NOR gates and a 74LS139 containing 2 2-to-4 decoders (A and B). The address lines SA2-SA9 constitute an 8-bit input byte compared by the 74LS688 to a second byte determined by an 8-fold dipswitch. This selects the base of the 4-byte I/O address space of the card. It can be \$0, \$4, \$8, \$C etc. The addresses \$300-\$31F are reserved for a prototype card. The base address \$300 (binary 11,0000,0000) is selected with bits 1100,0000 for connections 1,2,. . .,8 of the dip-switch. In the remainder this base address will be assumed. The enable input $\overline{\mathrm{G}}$ of the comparator is connected to the AEN line. This accomplishes that the card is not selected during DMA transfers.

On selection of one of the addresses \$300 . . . \$303 the 74LS688 output $\overline{\mathrm{P}=\mathrm{Q}}$ becomes low. This line is input to the two NOR gates A and B of the 74LS02. The truth table of a NOR gate is:

X	Y	$\overline{\mathrm{X}+\mathrm{Y}}$
0	0	1
0	1	0
1	0	0
1	1	0

Table 1: NOR gate truth table

A NOR gate with X=Y acts like an inverter. It follows that the enable $\overline{\mathrm{G}}$ of the A-part (B-part) of the decoder 74LS139 is low when both inputs of 74LS02A(B) are low. This happens when one of the addresses \$300 . . . \$303 is selected, while at the same time $\overline{\mathrm{IOR}}$ ($\overline{\mathrm{IOW}}$) is low. Because SA0 and SA1 are connected to the A and B inputs of the decoders, $\overline{\mathrm{Y0}}, \overline{\mathrm{Y1}}, \overline{\mathrm{Y2}}$ or $\overline{\mathrm{Y3}}$ of the 74LS139A(B) will become low during an I/O read (write) operation on \$300 . . . \$303, respectively. So only one of the eight decoder outputs is low at a time. The remaining seven output are high.

1.3 AD7569 converter

The AD7569 contains an ADC and a DAC. The jumpers JP1 and JP2 are used to select the range and data format of both converters, according to the next table.

JP1 (RANGE)	JP2 (V_{ss})	voltage range	DB0-DB7 data format
0 V	0 V	$0 \rightarrow +1.25$ V	binary
5 V	0 V	$0 \rightarrow +2.5$ V	binary
0 V	-5 V	$-1.25 \rightarrow +1.25$ V	2s complement
5 V	-5 V	$-2.5 \rightarrow +2.5$ V	2s complement

Table 2: Selectable voltage ranges and data formats

The inverted RESET of the XT bus is connected to the AD7569's $\overline{\mathrm{RESET}}$ input. The clock (CLK) is connected to an RC circuit, enabling the conversion frequency to be adjusted with the variable 10K resistor. A value of 6.8 kΩ corresponds to an AD conversion time of ≈ 2 μs. The analog input and output voltage are connected to a 9-pin hooked SUB-D connector.

1.3.1 DA conversion

The 74LS08A AND port enables the AD7869 by making input $\overline{\mathrm{CS}}$ low as soon as a read or write takes place on address \$301. A DA conversion is initiated by writing a byte to \$301, because $\overline{\mathrm{Y1}}$ of the 74LS139A decoder is tied to the $\overline{\mathrm{WR}}$ line of the converter.

1.3.2 AD conversion

Reading a byte from \$301 starts an AD conversion, because the $\overline{\mathrm{RD}}$ and $\overline{\mathrm{ST}}$ (start conversion) lines become low simultaneously. The byte obtained from the next read (more than $\approx 2\ \mu$s later) is the conversion result, and a new conversion is started immediately. This works out because the result of an AD conversion is stored in a latch on the AD7569. Only after completion of the next conversion the latched result is replaced by the new result.

The least significant bit of the byte read from \$300 is the by the 74LS240B inverted $\overline{\mathrm{BUSY}}$ signal of the AD7569. So bit 0 of address \$300 becomes 0 upon completion of an AD conversion. The 3-state 74LS240B is enabled (puts its data on the bus) as soon as a read from \$300 takes place.

2 Construction

I built the circuit on a double sided full length card for an XT slot (62 contacts), with 3-hole copper patches. A card like this, eventually including buffering and address decoding circuitry, is also called a prototype card. Included in the ADDA10.ZIP archive is a board layout sketch, showing the approximate IC positions (if I should build the board a second time, I would shift the whole setup to the right, creating more space for input signal shaping circuitry close to the connector). I wire-wrapped the board. Because the holes in the card were too large for standard wire-wrap pins I used parts of 36-pins rectangular SIL blocks. The circuit occupies about one third of the card space, leaving plenty of room for the addition of a programmable amplifier, a multiplexer or analog signal shaping circuitry (e.g., filters). These additional components can be programmed from the unused I/O addresses $302 and $303.

To minimize noise, correct grounding is important. The AD7569 has one digital ground (DGND), and two analog ones ($AGND_{DAC}$ and $AGND_{ADC}$). These have to be connected to each other as close to the IC as possible. The common connection is called "STAR GND" in the diagram. This prevents the fluctuating supply current from influencing the $AGND_{DAC}$ and $AGND_{ADC}$ grounds. Other connections to the analog grounds are made directly to the pins. It is possible that using IC's from the HCT instead of the LS series reduces noise because of the much reduced power consumption.

The 5 V power supply is decoupled to ground in some places with 100 nF ceramic capacitors, and all unused inputs are tied low.

2.1 Component list

Resistors:	8-fold 10K, 2 × 10K, potmeter 10K
Capacitors:	68 pF, 4 × 100 nF (ceramic)
Digital IC's:	74LS02, 74LS08, 74LS139, 74LS240, 74LS244, 74LS245, 74LS688
Linear IC's:	AD7569 JN
IC sockets:	2 × 14-pins, 16-pins, 4 × 20-pins, 24-pins
Miscellaneous:	8-fold dip-switch, 6 × 36-pins rectangular SIL block, 2 × 3-pins jumper, 9-pins hooked SUB-D connector, 1 × prototype card, wire-wrap wire

I paid about 100 US$ for the complete board (prototype card 30 US$, AD7569 20 US$, SIL blocks 20 US$, remaining parts 30 US$). Soldering instead of wire-wrapping saves the costs of the SIL blocks, reducing the costs to 80 US$. The prices are computed with an exchange rate of 1.70 Dutch guilders per US$.

3 Programming examples

All the figures quoted below were obtained from measurements with a 33 MHz 80386 based PC with its ISA (AT) I/O bus running at 11 MHz. The card was operated with the -1.25 $\rightarrow$ 1.25 V range. The corresponding 2s complement format (see Table 2) is the same as that of a Turbo Pascal ShortInt.

The programming examples are taken from the Turbo Pascal 6.0 source ADDA.PAS included in the ADDA10.ZIP archive. The part defining global constants and variables is:

```
program ADDA(Input, Output);

{ This program is to be used in conjunction with the analog extension
  board described in the note "ANALOG I/O BOARD FOR PC'S". It is assumed
  that the AD7569 is operated in one of its two bipolar modes (-1.25 V -->
  +1.25 V or -2.5 V --> +2.5 V). }

uses Dos, Crt, Graph;

const Title= 'ADDA V1.0 - Jos Groot (September 19, 1992)';

{$R-} { this prevents Turbo Pascal from complaining about putting a Byte
        into a ShortInt type variable }

const Status = $300 ;  { bit 0 is 1 when an AD conversion is going on and
                        becomes 0 upon completion of the conversion }
      ADC    = $301 ;  { read : get result of most recently completed AD
                        conversion and start the next }
      DAC    = ADC  ;  { write: start DA conversion }
      Max_Sam= 60000;  { maximum number of samples }

var Buf: array[0..Max_Sam] of ShortInt; { global array receiving samples }
```

3.1 Basic use

The card is very simple to use as can be seen from the next procedure which takes a number of samples:

```
procedure Get_Samples(Samples: Word);

{ reads as fast as possible samples 0,1,...,Samples into array Buf. Sample
  Buf[0] should not be used, because this is the ADC result of a conversion
  started at an unknown earlier time. }

var i: Word;
    Pause: Integer;
```

```
begin
  for i:= 0 to Samples do { start at 0, 1 is the first good sample to use }
  begin
    asm   { small delay to enable the ADC to complete a conversion before  }
      nop { reading the result. The necessity and length of this delay may }
      nop { vary for different AD7569's. }
    end;

{   for Pause:= 1 to 50 do; } { add this loop to decrease sampling frequency }

    Buf[i]:= Port[ADC] { get and store sample & initiate next conversion }
  end
end;
```

This procedure accomplishes a sample frequency of about 440 kHz (the AD7569 datasheets guarantee the maximum sampling frequency to be between 385 and 625 KHz with the internal clock, and at least 500 kHz with an external clock). In still faster systems or machinecode routines one can test bit 0 of port $300 to check for conversion completion. During conversion this bit is 1, upon completion it becomes 0.

A procedure to copy the ADC input to the DAC is:

```
procedure ADC_To_DAC;

{ reroutes ADC input directly to DAC output }

begin
  repeat
    Port[ADC]:= Port[DAC]
  until KeyPressed
end;
```

3.2 Digital guitar effects

I used the following setup for experiments with real time digital signal processing to produce the 'guitar effects' distortion/compression and echoing:

electric guitar → pre-amplifier → ADC input → PC + Turbo Pascal program → DAC output → guitar amplifier

Care should of course be taken that the maximum ratings of the ADC, the guitar amplifier input etc. are not exceeded. The procedures are:

```
procedure Distortion;

{ reads ADC values, and outputs 100 to the DAC for the ones with absolute
```

```
  value>= Clip_Level. This produces a compressed and heavily distorted
  sound. }

const Clip_Level= 10; { should at least exceed the maximum value of absolute
                        noise samples }

var s: ShortInt;
    Table: array[-128..127] of ShortInt; { lookup table for fast execution }

begin
  for s:= -128 to 127 do
    if s>  Clip_Level then Table[s]:= 100 else
    if s< -Clip_Level then Table[s]:= 100 else { -100 for softer distortion }
                           Table[s]:= Abs(Round(s/Clip_Level*100));

  repeat
    Port[DAC]:= Table[ShortInt(Port[ADC])]
  until KeyPressed
end;

{ ******* }

procedure Echo;

{ produces an echo by adding ADC samples from some time ago to the present
  samples, and outputting the result to the DAC }

var i, j, d: Word;
    k, s, Pause, Buffers: Integer;
    Table: array[-128..127] of Integer; { lookup table for fast execution }
    Dr: Real;

begin
  Pause   := 20   ; { Pause determines the sampling frequency (40 kHz)       }
  d       := 20000; { d and Pause determine the delay time (20E3/40E3=0.5 s) }
  Dr      := 0.5  ; { Dr is inversely proportional to the decay rate         }
  Buffers:= 0     ; { number of Max_Sam byte Buffers processed               }

  for i:= 0 to Max_Sam do Buf[i]:= 0;            { clear buffer }
  for k:= -128 to 127 do Table[k]:= Round(Dr*k); { fill table }

  WriteLn; WriteLn; WriteLn;
  Write('Number of ', Max_Sam:1, ' byte buffers processed: ');

  repeat
    GotoXY(41,16);    { print the number of processed buffers indicating }
```

```
    Write(Buffers:1); { the sampling frequency }
    Inc(Buffers);

    for i:= 0 to Max_Sam do
    begin
      for k:= 1 to Pause do;                       { lower sampling frequency }
      if i>=d then j:= i-d else j:= i-d+Max_Sam+1; { j= index delayed sample }
      s:= ShortInt(Port[ADC]) + Table[Buf[j]];     { compute original + echo }
      if s<-128 then s:= -128 else if s>127 then s:= 127; { correct overflow }
      Buf[i]:= s;                                  { store compound sample   }
      Port[DAC]:= s                                { output sample to DAC    }
    end
  until KeyPressed
end;
```

To get long delay times the loop creating a pause had to be included. The fact that the sampling frequency is still as high as 40 kHz indicates that more complex real time effects can be implemented on this kind of 80386 machines.

Some remarks about the noise and sound quality are in place here. The board is used in an electrically noisy environment. This, together with the large ADC input bandwidth causes 'noisy' ADC output values. Remember, when sampling at for example 40 kHz, all signals above 20 kHz manifest itself as noise in the ADC output. This is called aliasing. A bandwidth limiting input filter should be used to limit this kind of noise. A measure for the noise is the RMS (Root-Mean-Square) amplitude of the samples taken when only noise is input to the ADC. The RMS noise amplitude for the samples $s_1, s_2, s_3, \ldots, s_n$ is defined by

$$RMS_{noise} = \frac{1}{n}\sqrt{n\sum_{i=1}^{n} s_i^2 - \left(\sum_{i=1}^{n} s_i\right)^2} \tag{1}$$

The smaller the noise, the smaller this value. Without noise (all samples equal) the RMS noise is 0.

The RMS noise with the pre-amplifier volume set to zero is about 3-5. The figure depends on the AD7569 clock frequency, which can be adjusted to minimize the noise. By inserting a simple passive first-order (6 dB/octave) RC filter before the ADC input with a 10 kHz cutoff frequency (estimated, not measured), the RMS noise drops to well below 1.

The sound quality obtained with only 8 bits is better than I expected. Of course, more bits are better, but maybe the extra (least significant) bits are completely dominated by the noise when a wire-wrapped (non-ideally constructed) card like this is used. The board is in my opinion well suited for exploring the basics of real time digital signal/sound processing. For stage sound quality one should buy a dedicated effects processor.

4 Excerpts from the AD7569 data sheets

The AD7569 is a complete, 8-bit, analog I/O system on a single monolithic chip. The AD7569 contains a high speed successive approximation ADC with 2 μs conversion time, a track/hold with 200 kHz bandwidth, a DAC and output buffer amplifier with 1 μs settling time. A temperature- compensated 1.25 V reference provides a precision reference voltage for the ADC and the DAC.

A choice of analog input/output ranges is available. Using a supply voltage of +5 V, input and output ranges of zero to 1.25 V and zero to 2.5 V may be programmed using the RANGE input pin. Using a ±5 V supply, bipolar ranges of ±1.25 V or ±2.5 V may be programmed.

Digital interfacing is via an 8-bit I/O port and standard microprocessor control lines. Bus interface timing is extremely fast, allowing easy connection to all popular 8-bit microprocessors. A separate start convert line controls the track/hold and ADC to give precise control of the sampling period.

The AD7569 provides everything necessary to interface a microprocessor to the analog world. No external components or user trims are required, and the overall accuracy of the system is tightly specified, eliminating the need to calculate error budgets from individual component specifications.

Pin function description (the electronic diagram shows the pin numbers):

$\mathbf{AGND_{DAC}}$ Analog ground for the DAC. Separate ground return paths are provided for the DAC and ADC to minimize crosstalk.

$\mathbf{V_{OUT}}$ Output voltage. V_{OUT} is the buffered output voltage from the AD7569 DAC. Four different output voltage ranges can be achieved (see Table 2).

$\mathbf{V_{SS}}$ Negative supply voltage (-5 V for dual supply or 0 V for single supply). This pin is also used with the RANGE pin to select the different input/output ranges and changes the data format from binary (V_{SS}=0 V) to 2s complement (V_{SS}=-5 V) (see Table 2).

RANGE Range selection input. This is used with the V_{SS} input to select the different ranges as per Table 2. The range selected applies to both the analog input voltage of the ADC and the output voltage from the DAC.

$\overline{\mathbf{RESET}}$ Reset input (active low). This is an asynchronous system reset which clears the DAC register to all 0s and clears the $\overline{\text{INT}}$ line of the ADC (i.e., makes the ADC ready for new conversion). In unipolar operation this input sets the output voltage to 0 V; in bipolar operation it sets the output to negative full scale.

DB0-DB7 Data bits. DB0 is the least significant bit.

DGND Digital ground.

$\overline{\mathbf{WR}}$ Write input (edge triggered). This is used in conjunction with $\overline{\text{CS}}$ to write data into the AD7569 DAC register. Data is transferred on the rising edge of $\overline{\text{WR}}$.

$\overline{\mathbf{CS}}$ Chip select input (active low). The device is selected when this input is active. DB0-DB7 are in the high impedance state when this input is high.

$\overline{\mathbf{RD}}$ Read input (active low). This input must be active to access data from the part. It is used in conjunction with the $\overline{\mathrm{CS}}$ input.

$\overline{\mathbf{ST}}$ Start conversion (edge triggered). This is used when precise sampling is required. The falling edge of $\overline{\mathrm{ST}}$ starts conversion and drives $\overline{\mathrm{BUSY}}$ low. The $\overline{\mathrm{ST}}$ signal is not gated with $\overline{\mathrm{CS}}$

$\overline{\mathbf{BUSY}}$ BUSY status output (active low). When this pin is active the ADC is performing a conversion. The input signal is held prior to the falling edge of $\overline{\mathrm{BUSY}}$.

$\overline{\mathbf{INT}}$ INTERRUPT output (active low). $\overline{\mathrm{INT}}$ going low indicates that the conversion is complete. $\overline{\mathrm{INT}}$ goes high on the rising edge of $\overline{\mathrm{CS}}$ or $\overline{\mathrm{RD}}$ and is also set high by a low pulse on $\overline{\mathrm{RESET}}$.

CLK A TTL compatible clock signal may be used to determine the ADC conversion time. Internal clock operation is achieved by connecting a resistor and capacitor to ground.

$\mathbf{AGND_{ADC}}$ Analog ground for the ADC.

$\mathbf{V_{IN}}$ Analog input. Various input ranges can be selected (see Table 2).

$\mathbf{V_{DD}}$ Positive supply voltage (+5 V).

Absolute maximum ratings:

V_{DD} to $AGND_{DAC}$ or $AGND_{ADC}$	-0.3 V, +7 V
V_{DD} to DGND	-0.3 V, +7 V
V_{DD} to V_{SS}	-0.3 V, +14 V
$AGND_{DAC}$ or $AGND_{ADC}$ to DGND	±5 V
Logic voltage to DGND	-0.3 V, V_{DD}+0.3 V
CLK input voltage to DGND	-0.3 V, V_{DD}+0.3 V
V_{OUT} to $AGND_{DAC}$[1]	$V_{SS} - 0.3V,\ V_{DD} + 0.3V$
V_{IN} to $AGND_{ADC}$	$V_{SS} - 0.3V,\ V_{DD} + 0.3V$
Power dissipation	450 mW

[1]The output may be shorted to any voltage in the range V_{SS} to V_{DD} provided that the maximum power dissipation (450 mW) of the package is not exceeded. Typical short circuit current for a short to AGND or V_{SS} is 50 mA.

I did my utmost best to exclude all errors from the contents of the ADDA10.ZIP archive. Because I am just like you only human, the described circuit might not work or, even worse, cause damage to other equipment. Nevertheless, I assume no liability for this. The only guarantee I can give you is that my board[2] functions up to expectation. Of course any suggestions to improve the correctness and intelligibility of the description, diagrams and software are much appreciated, as are experiences of other people who built and used the circuit. These will have their impact on possible future versions of the archive.

Jos Groot (InterNet: jos.groot@fel.tno.nl)

van de Wateringelaan 4
2274 CH Voorburg
The Netherlands

[2]I built the board with the electronic diagram included in this archive; I did not create the diagram after building the board