Programy shareware - płyta ANALOG I/O do PC

Tym razem jako oferta programu shareware'owego znalazła się rzecz bardzo nietypowa, doskonale natomiast pokrywająca się z profilem pisma. Użyłem sformułowania rzecz, a nie program, ponieważ nie o program tu chodzi lecz raczej o kit, z jakim macie Państwo „na codzień" do czynienia na łamach EP. Konkretnie mamy do zaoferowania pakiet rozprowadzany w formie shareware, który w sumie tworzy kompletną dokumentację karty przetwornika DA/AD do komputera klasy PC wraz z programem (z kodem źródłowym w Turbo Pascalu) obsługującym tę kartę.

Jos Groot
jos.groot@fel.tno.nl
van de Wateringelaan 4
2274 CH Voorburg
Holandia

W skład tego pakietu wchodzi: dokumentacja (formaty: dvi, hp, tex, ps), rozmieszczenie elementów na płytce (formaty: hp, ps), schemat (formaty: ps, hp), program wraz z kodem źródłowym w Turbo Pascalu 6.0.

Karta jest zbudowana na układzie AD7569 zawierającym 8-mio bitowy przetwornik DA 1µs i 8-mio bitowy przetwornik AD 2µs. Może on służyć na przykład do „samplowania" i odtwarzania, po ewentualnym uprzednim przetworzeniu, wprowadzonego sygnału.

Dokumentacja do układu (niestety w języku angielskim) zawiera:
- dokładny opis działania układu wraz z opisem konwertera AD7569;
- konstrukcję układu ze spisem elementów;
- przykłady programów - użycie podstawowe, efekty gitarowe distortion i echo itp.- napisane w Turbo Pascalu 6.0;
- wyjątki z karty katalogowej układu AD7569.

Dołączony program (zarówno w wersji *.exe jak i kodu źródłowego w Turbo Pascalu 6.0) umożliwia:
- oglądanie na ekranie przebiegów sygnałów;
- wykonywanie efektów echa, distortion;
- proste sterowanie kartą.

Program jest napisany bardzo przejrzyście i jest doskonałą podstawą dla pisania własnych procedur i kompletnych programów.

Użyte układy scalone: 74LS02, 74LS08, 74LS240, 74LS139, 74LS244, 74LS 245, 74LS688, AD7569.

Paweł Marciniak

```
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;

                          Table[s]:= Abs(Round(s/Clip_Level*100));

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

Przykład podstawowego programu (zaczerpnięty z dołączonej dokumentacji)

```
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 }
end
```

Przykład prostego efektu gitarowego DISTORTION (zaczerpniqty z dołączonej dokumentacji)