MODUL 2

# Codec dan Sampling

## 1. Pendahuluan

DSP mengolah data dalam dunia digital (sinyal diskrit) seperti namanya yang terdengar indah, *Digital* Signal Processor. Sedangkan kita hidup di dunia analog (sinyal kontinyu) dengan sinyal-sinyal seperti bunyi atau suara, suhu, cahaya, tegangan listrik dan arus listrik. Kemudian bagaimana cara mengubah sinyal analog menjadi sinyal digital agar suara (misalnya) dapat diproses oleh DSP dan mengubah kembali sinyal digital hasil proses di DSP menjadi sinyal analog yang bisa kita dengarkan? Terima kasih kepada penemu ADC dan DAC. ADC mengkonversi analog menjadi digital dan DAC mengkonversi digital ke analog. Kadang Anda akan menemukan ADC dan DAC didalam sebuah kemasan IC, biasa disebut sebagai IC codec. Istilah codec merupakan kepanjangan dari coder-decoder. Coder adalah ADC dan decoder adalah DAC. Jadi didalam sebuah IC codec, terdapat ADC dan DAC sekaligus. ADC bekerja dengan cara melakukan sampling atau pencuplikan sinyal pada kecepatan tertentu. Hubungan antara kecepatan sampling (juga dapat disebut sebagai frekuensi sampling) dengan frekuensi sinyal input yang masuk akan anda pelajari pada praktikum ini. Sehingga anda akan segera dapat menentukan bahwa tidak mungkin untuk membaca sinyal pada frekuensi 1MHz dengan ADC yang bekerja pada frekuensi sampling 48KHz.

## 2. Tujuan

Setelah menyelesaikan praktikum ini, yang anda peroleh adalah :

- dapat menjelaskan pengaruh frekuensi sampling pada saat melakukan konversi sinyal analog ke digital
- dapat menggunakan fasilitas audio codec pada board DSK

## 3. Gambaran Disain

Board DSK yang anda gunakan dilengkapi dengan IC codec. Anda dapat melihat terminal berupa jack untuk audio-in (mic) bersebelahan dengan jack untuk audio-out (speaker) yang terhubung dengan IC codec. Pada praktikum ini anda akan mencoba untuk mengambil data pada audio-in dan mengeluarkan kembali data pada audio-out dengan cara mengakses IC codec. Frekuensi sampling dan penguatan (gain) dari IC codec dapat diprogram secara software. Disini anda akan membuktikan pengaruh frekuensi sampling terhadap frekuensi sinyal input yang masuk pada jack audio-in. Sebagai sinyal input digunakan function generator dan outputnya dilihat menggunakan oscilloscope.

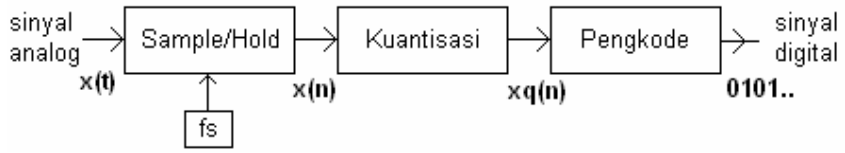
# 4. Dasar Teori

Sebuah sinyal mengandung informasi tentang amplitudo, frekuensi dan sudut fasa. Pengolahan sinyal biasanya digunakan untuk mendapatkan informasi dari sebuah sinyal.

Mendapatkan informasi dari sebuah sinyal menggunakan perangkat analog adalah rumit dan kurang akurat. Karena itu kita gunakan metode pengolahan yang lebih sederhana, fleksibel dan akurat, yaitu pengolahan sinyal digital (DSP).

Untuk pengolah sinyal analog dengan perangkat digital, yang pertama dilakukan adalah mengubah sinyal analog menjadi sederetan angka yang mempunyai keakuratan tertentu. Langkah ini disebut konversi analog ke digital, menggunakan alat yang disebut ADC (Analog to Digital Converter). Supaya sinyal digital ini cukup akurat untuk dikembalikan lagi menjadi sinyal analog maka perlu diperhatikan masalah jumlah sampling yang dipilih oleh ADC dan besarnya angka yang dipakai untuk mewakili tiap sampling. Teori sampling membantu kita untuk menentukan jumlah sampling yang diperlukan untuk menghasilkan kembali sinyal analog berdasarkan frekuensi maksimum pada sinyal analog yang diolah.

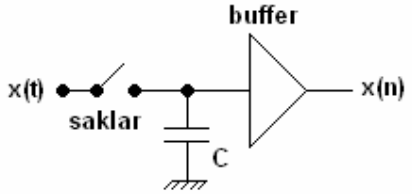
Blok diagram dasar dari sebuah ADC ditunjukkan oleh gambar 1.



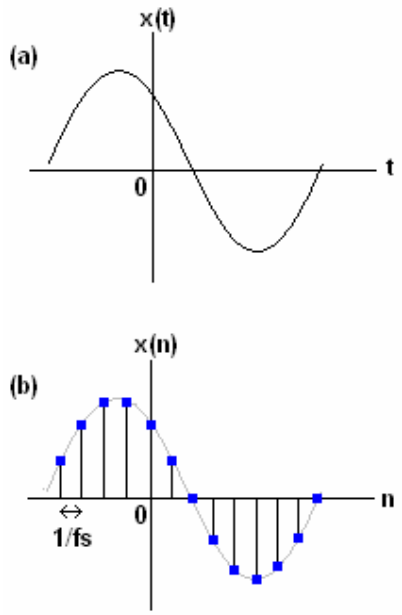

Gambar 1. Proses konversi sinyal analog menjadi sinyal digital

## Sampling

Proses pencuplikan secara sederhana ditunjukkan oleh gambar 2. Apabila saklar ditutup sebentar kemudian dibuka kembali maka kapasitor C akan terisi muatan yang sama dengan besar sinyal x(t) saat saklar ditutup. Buffer ditambahkan agar muatan kapasitor tetap terjaga saat digunakan oleh proses berikutnya. Perhatikan saklar, apabila saklar ditutup dan dibuka dengan kecepatan tetap sebesar fs maka akan didapatkan titik-titik yang berjarak sama seperti yang ditunjukkan oleh gambar 3.




Gambar 2. Pencuplikan sinyal

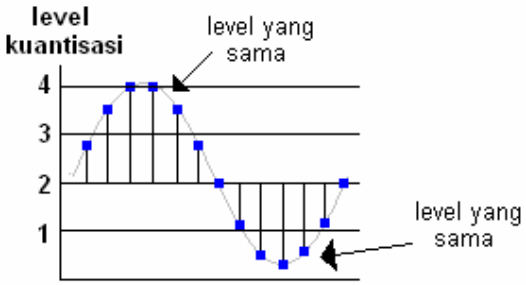



Gambar 3. Proses pencuplikan. (a) Sinyal analog. (b) Hasil sinyal yang dicuplik

## Kuantisasi

Sinyal digital merupakan sebuah deretan angka (sampling) yang diwakili oleh beberapa digit dengan jumlah tertentu (menentukan keakuratan).

Proses melakukan konversi sinyal yang telah dicuplik menjadi sinyal digital yang diwakili oleh sebuah nilai dengan jumlah digit tertentu disebut kuantisasi.




Gambar 4. Proses kuantisasi

Gambar 4 adalah contoh proses kuantisasi yang menggunakan empat level. Anda dapat melihat pada level 4 terdapat empat buah sinyal yang menempati level yang sama, artinya keempat sinyal tersebut dikelompokkan menjadi level yang sama walaupun tingginya berbeda. Demikian pula pada level 1. Selisih antara nilai kuantisasi dengan sinyal sebenarnya disebut kesalahan kuantisasi (error quantization). Maka

$$e_q(n) = x_q(n) - x(n)$$

Jarak antara level kuantisasi disebut resolusi. Kuantisasi merupakan proses yang tidak dapat dibalik sehingga menyebabkan distorsi sinyal yang tidak dapat diperbaiki.

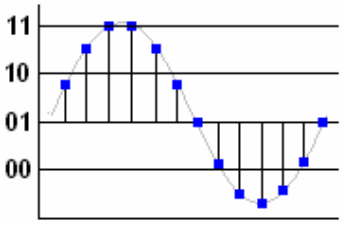
Pada gambar 4, untuk mengurangi kesalahan kuantisasi, dengan kata lain agar ADC mempunyai ketelitian yang tinggi maka resolusi harus ditingkatkan. Memperbanyak level kuantisasi.

Jadi harus berapa level? 8? 10? 12? 24? Bila anda mendisain sebuah ADC anda juga harus mempertimbangkan faktor harga dan aplikasi yang akan menggunakan.

## Pengkode

Proses pengkodean dalam ADC menetapkan bilangan biner tertentu pada tiap level kuantisasi. Bila kita mempunyai level kuantisasi sejumlah L, maka kita membutuhkan bilangan biner paling tidak sejumlah L. Anda membutuhkan digit yang diperlukan sebanyak b-bit sehingga $2^b \geq L$.

Untuk gambar 4, terdapat empat level kuantisasi sehingga dibutuhkan 2-bit saja. Jadi kode biner untuk gambar 4 adalah 00, 01, 10, 11.




Gambar 5. Proses pengkodean

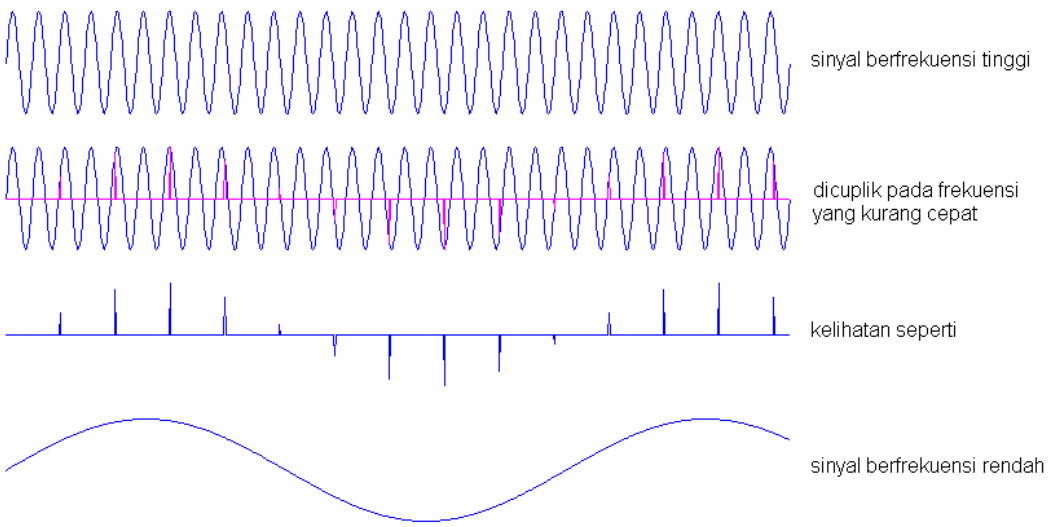
Sinyal pada gambar 5 setelah keluar dari ADC akan mempunyai kode biner 10,11,11,11,11,10,01,01,00,00,00,01,01

## Teori Sampling

Kecepatan pengambilan sampel (frekuensi sampling) dari sinyal analog yang akan dikonversi haruslah memenuhi kriteria Nyquist yaitu:

$$Fs > 2\, F_{in}max$$

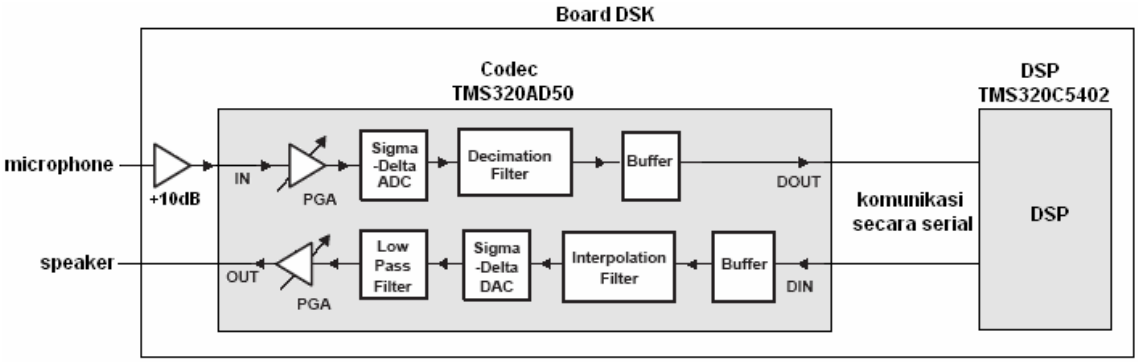
dimana frekuensi sampling (Fs) minimum adalah 2 kali frekuensi sinyal analog yang akan dikonversi ($F_{in}$max). Misalnya bila sinyal analog yang akan dikonversi mempunyai frekuensi sebesar 50Hz maka frekuensi sampling minimum dari ADC adalah 100Hz. Atau bila dibalik, bila frekuensi sampling ADC sebesar 100Hz maka sinyal analog yang akan dikonversi harus mempunyai frekuensi maksimum 50Hz. Apabila kriteria Nyquist tidak dipenuhi maka akan timbul efek aliasing yang diilustrasikan oleh gambar 3. Disebut aliasing karena frekuensi tertentu terlihat sebagai frekuensi yang lain (menjadi alias dari frekuensi lain).




Gambar 3. Aliasing

Codec merupakan kepanjangan dari coder-decoder. Coder bertugas untuk mengubah sinyal analog menjadi sinyal digital dan dekoder bertugas untuk mengubah sinyal digital menjadi sinyal analog kembali. Codec secara sederhana merupakan sebuah ADC-DAC yang dikemas dalam sebuah chip.

Pada DSK, pekerjaan codec dilakukan oleh IC TLC320AD50. Gambaran audio codec pada DSK secara sederhana ditunjukkan oleh gambar 4. Sinyal masukan audio maksimum yang diijinkan sebesar 500mV. Pada board DSK terdapat preamplifier dengan penguatan 10dB sebelum memasuki IC codec. IC codec ini mempunyai PGA (Programmable Gain Amplifier) yang dapat diatur secara software. Sinyal masukan audio dapat dikuatkan dari +0dB sampai +12dB dengan step 6dB. Sedangkan sinyal keluaran audio dapat dilemahkan dari +0dB sampai -12dB dengan step 6dB.




Gambar 4. Ilustrasi bagian codec pada DSK yang disederhanakan

Berikut sedikit keterangan tentang bagian-bagian didalam IC codec TLC320AD50. ADC didalam codec menggunakan *modulator sigma-delta* dengan 64x *oversampling*. Dengan teknik oversampling ini ADC mempunyai sifat resolusi tinggi dan rendah noise. *Decimation filter* menurunkan kecepatan data digital dari ADC pada kecepatan sampling dengan cara diturunkan dengan perbandingan 1:64. Data output dari decimation filter mempunyai lebar

data 16-bit dalam format *2's complement* yang dikeluarkan pada kecepatan sampling. Frekuensi cut-off dari filter decimation adalah 0.439 x $f_{sampling}$ dan terskala secara linier terhadap frekuensi sampling yang digunakan. DAC yang digunakan didalam codec menggunakan modulator sigma-delta dengan 256x oversampling agar DAC mempunyai unjuk kerja dengan resolusi tinggi dan rendah noise. *Interpolation filter* mencuplik ulang data digital yang diterima pada kecepatan 256 kali dari kecepatan data yang diterima. Data berkecepatan tinggi ini akan digunakan oleh DAC. Frekuensi cut-off dari filter interpolasi adalah 0.439 x $f_{sampling}$ dan terskala secara linier terhadap frekuensi sampling yang digunakan. Pusing ya? Lebih detil tentang kata-kata tercetak miring diatas dapat anda pelajari dari literatur lain.

## Referensi:


- Tutorial Code Composer Studio 2.
- Datasheet TLC320AD50C/I, slas131e, Texas Instruments, 2000


## 5. Peralatan

- 1 set PC yang dilengkapi dengan software Code Composer Studio.
- 1 set DSK TMS320C5402
- 1 set function generator
- 1 set oscilloscope

## 6. Prosedur Praktikum

Anda diharapkan mengikuti langkah-langkah prosedur praktikum dan apabila ada kesulitan harap bertanya kepada asisten.

Pada praktikum ini anda akan mengamati dua hal, yaitu:

- 6.1. Pengaruh penguatan sinyal sebelum dan sesudah melewati codec pada board DSK
- 6.2. Pengaruh kecepatan pencuplikan (sampling rate) codec terhadap sinyal input

Praktikan harus didampingi Asisten ketika menjalankan bagian ini !!!

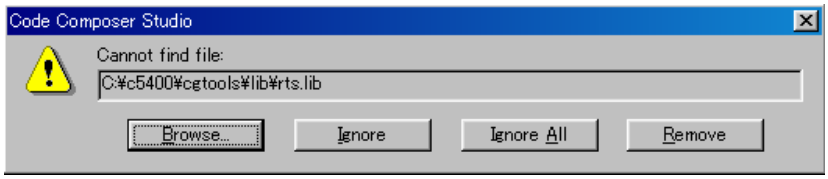
### *6.1. Penguatan codec pada board DSK*

1. Menyiapkan peralatan :
   a. PC dalam keadaan mati.
   b. Hubungkan DSK ke PC menggunakan kabel paralel port yang tersedia.
   c. Hubungkan output adaptor ke input power DSK.
   d. Hubungkan kabel power adaptor, nyalakan adaptor.
   e. Nyalakan PC.

f. Jalankan aplikasi Code Composer Studio dan pastikan dapat terhubung dengan board DSK.

2. Dengan menggunakan Windows Explorer, buatlah folder baru pada direktori **D:\prak_pengolahansinyal** dengan kelas Anda diikuti dengan subfolder nama Anda. Perhatikan penulisan folder yang Anda buat. Kemudian salinlah direktori codec pada **C:\ti\examples\dsk5402\dsp\codec** kedalam direktori **D:\prak_pengolahansinyal*kelas**nama***. Hal ini dimaksudkan untuk mempermudah mengembalikan isi project seperti dalam keadaan semula apabila terjadi kesalahan.

3. Pada Code Composer Studio (CCS), dengan menggunakan **Project → Open**, bukalah file project *codec.pjt* pada direkori **D:\prak_pengolahansinyal*kelas**nama*\codec**. Apabila ada file library pada project tersebut yang tidak ditemukan maka carilah library tersebut pada direktori c:\ti yang sesuai. Hal ini terjadi karena lokasi project berpindah tempat. File library yang digunakan ada tiga yaitu:
   a. rts.lib pada direktori c:\ti\c5400\cgtools\lib
   b. dsk5402.lib pada direktori c:\ti\c5400\dsk5402\lib
   c. drv5402.lib pada direktori c:\ti\c5400\dsk5402\lib




Gambar 5. File library tidak ditemukan.

4. Pilih **Project → Rebuild All**, maka akan timbul pesan kesalahan, 1 error dan 4 warning. Klik dua kali pada window build di bagian bawah, scroll ke atas untuk melihat pesan error, klik dua kali pada pesan error *"codec.c", line 8: fatal error: could not open source file "type.h"* maka listing codec.c baris ke-8 akan ditampilkan oleh kursor yang berkedip.

5. Pilih **Project → Build Option**, pada tab Compiler, pilih *preprocessor* pada Category. Pada kolom *Include Search Path* ketikkan: c:\ti\c5400\dsk5402\include, klik tombol OK. Kemudian rebuild kembali dengan memilih Project → Rebuild All. Proses kompilasi berhasil dilakukan. Pesan kesalahan berupa *warning* dapat diabaikan.

6. Pilih **File → Load Program**, browse pada direktori debug, pilih file codec.out. Maka CCS akan meload program pada target DSP dan membuka window dis-assembly yang memperlihatkan instruksi program dalam bahasa assembler.

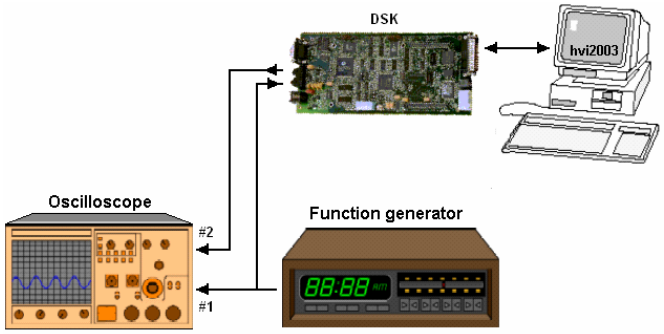
7. Pilih **Debug → Go Main**

8. Siapkan oscilloscope yang telah dikalibrasi. Atur volt/div pada 0.5 volt dan atur time/div pada 1ms.

9. Siapkan function generator untuk menghasilkan sinyal sinusoida dengan frekuensi 1KHz dan amplitudo 50mV.

Bila d*isplay* pada function generator menunjukkan amplitudo sinyal 50mV, maka amplitudo sinyal sebenarnya adalah 100mVpp (tegangan puncak ke puncak)

10. Hubungkan function generator, oscilloscope dan board DSK seperti ditunjukkan oleh gambar 6. Pada function generator gunakan konektor-T.




Gambar 6. Hubungan function generator, oscilloscope dan board DSK

11. Pilih **Debug → Run**, amati display pada oscilloscope.

12. Pilih **Debug → Halt**, untuk menghentikan eksekusi pada board.

13. Kerjakan tugas ke-1.

### *6.2. Frekuensi sampling (Sampling rate) pada codec*

1. Kembalikan nilai gain untuk input analog dengan nilai CODEC_AIN_0dB dan gain untuk output analog dengan nilai CODEC_AOUT_MINUS_6dB. Kemudian lakukan **rebuild** kembali project tersebut.

2. Pilih **File → Load Program**, browse pada direktori debug, pilih file codec.out. Maka CCS akan meload program pada target DSP dan membuka window dis-assembly yang memperlihatkan instruksi program dalam bahasa assembler.

3. Dengan hubungan masih seperti gambar 6, atur kembali function generator untuk menghasilkan sinyal sinusoida dengan frekuensi 1 KHz dan amplitudo 50 mV.

4. Pilih **Debug → Run.**

5. Ubah frekuensi sinyal sinusoida semakin besar perlahan-lahan, pada frekuensi tertentu amati apa yang terjadi pada layar oscilloscope.

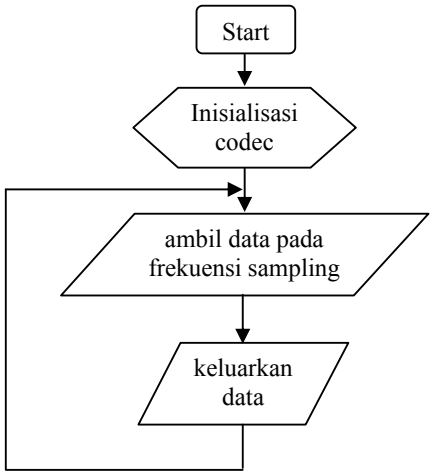
6. Hentikan eksekusi, pilih **Project → Halt.**

7. Kerjakan tugas ke-2

## 7. Tugas

1. Anda akan mencoba mengubah-ubah penguatan codec dengan cara mengatur bagian Programmable Gain Amplifier secara software (lihat kembali gambar 4). Ikuti langkah-langkah berikut:

   a. Buka file codec.c, flowchart dari program codec.c sebagai berikut :




   Pada bagian inisialisasi codec, anda dapat menentukan nilai penguatan (gain) dari codec. Cuplikan bagian inisialisasi codec sebagai berikut :

```
/* Open Handset Codec */
hHandset = codec_open(HANDSET_CODEC);
/* Set codec parameters */
codec_dac_mode(hHandset, CODEC_DAC_15BIT);
codec_adc_mode(hHandset, CODEC_ADC_15BIT);
codec_ain_gain(hHandset, CODEC_AIN_6dB);          /* ubah disini */
codec_aout_gain(hHandset, CODEC_AOUT_MINUS_6dB); /* ubah disini */
codec_sample_rate(hHandset,SR_16000);
```

   Penguatan pada input analog dapat diubah dengan memberikan nilai pada parameter kedua pada sub rutin `codec_ain_gain`, sedangkan gain pada output analog dapat diubah dengan memberikan nilai pada parameter kedua pada sub rutin `codec_aout_gain`.

   b. Atur kembali function generator untuk menghasilkan sinyal sinusoida dengan frekuensi 1 KHz dan amplitudo 50mV.

   c. Ubahlah nilai gain pada listing program codec.c dengan menggantikan parameter kedua seperti pada Tabel 1. Ada enam kali percobaan, setelah mengganti dengan konstanta baru lakukan rebuild kemudian download dan running, tuliskan hasil pengamatan anda kedalam tabel 1.

Tabel 1. Pengamatan penguatan codec dengan Vin=50mV (100mVp-p)

| No. | Konstanta pengganti untuk | | Vout (mVp-p) | Sinyal Cacat? (Y/T) |
|---|---|---|---|---|
| | Codec_ain_gain | Codec_aout_gain | | |
| 1 | CODEC_AIN_0dB | CODEC_AOUT_MINUS_0dB | | |
| 2 | | CODEC_AOUT_MINUS_6dB | | |
| 3 | | CODEC_AOUT_MINUS_12dB | | |
| 4 | CODEC_AIN_6dB | CODEC_AOUT_MINUS_0dB | | |
| 5 | CODEC_AIN_12dB | | | |
| 6 | CODEC_AIN_12dB | CODEC_AOUT_MINUS_12dB | | |

2. Berikutnya anda akan mencoba mengubah-ubah nilai dari frekuensi sampling codec. Ikuti langkah-langkah berikut:

   a. Buka file codec.c.

   b. Ubahlah sampling rate dari codec dengan mengubah nilai parameter kedua pada sub rutin codec_sample_rate didalam listing file codec.c.

```
/* Open Handset Codec */
hHandset = codec_open(HANDSET_CODEC);
/* Set codec parameters */
codec_dac_mode(hHandset, CODEC_DAC_15BIT);
codec_adc_mode(hHandset, CODEC_ADC_15BIT);
codec_ain_gain(hHandset, CODEC_AIN_6dB);
codec_aout_gain(hHandset, CODEC_AOUT_MINUS_6dB);
codec_sample_rate(hHandset,SR_16000);  /* ubah disini */
```

   c. Atur kembali function generator untuk menghasilkan sinyal sinusoida dengan frekuensi 1 KHz dan amplitudo 50mV.

   d. Ubahlah nilai frekuensi sampling pada listing program codec.c dengan menggantikan parameter kedua seperti pada Tabel 2 (menggantikan konstanta SR_16000 dengan SR_8000 kemudian dengan SR_4000). Setelah mengganti dengan konstanta baru lakukan rebuild kemudian download dan running, tuliskan hasil pengamatan anda kedalam tabel 2.

Tabel 2. Sampling rate pada codec

| No. | Vin = 50 mV (100mVp-p) | | | |
|---|---|---|---|---|
| | Frekuensi sinyal input (KHz) | Vout (mVp-p) untuk sampling rate : | | |
| | | SR_16000 | SR_8000 | SR_4000 |
| 1 | 1 | | | |
| 2 | 2 | | | |
| 3 | 3 | | | |
| 4 | 4 | | | |
| 5 | 5 | | | |
| 6 | 6 | | | |
| 7 | 7 | | | |
| 8 | 8 | | | |
| 9 | 9 | | | |

*Catatan :*
*Pada board DSK, anda tidak dapat melihat efek aliasing. Hal ini disebabkan karena codec yang digunakan pada board yaitu IC TLC320AD50 telah memiliki filter internal yang mempunyai frekuensi cut-off yang terskala secara linier mengikuti frekuensi sampling yang diberikan. Anda dapat meminta dosen atau asisten untuk mendemokan efek aliasing.*

e. Setelah selesai pilih **Project → Halt**.

f. Tutuplah program CCS dengan memilih **File → Exit**.

g. Matikan oscilloscope, function generator dan matikan PC dengan prosedur *shutdown* yang benar.

h. Matikan board DSK dengan cara mematikan adaptor, rapikan kabel-kabel dan peralatan yang telah anda gunakan.

## 8. Analisa

1. Amati hasil pengamatan pada Tabel 1. Mengapa terdapat hasil sinyal output yang cacat?
2. Amati hasil pengamatan pada Tabel 2. Bagaimana hubungan antara frekuensi sampling pada codec dengan frekuensi sinyal input?
3. Bagaimana cara untuk menghindari aliasing?

## 9. Pertanyaan pendahuluan

1. Sebuah sinyal dengan amplitudo 20 mV dimasukkan kedalam sistem yang mempunyai penguatan sebesar +10dB. Berapakah nilai tegangan outputnya?
2. Secara teori, apabila anda ingin menampilkan sinyal AC pada jala-jala listrik yang mempunyai frekuensi 50Hz, berapakah kecepatan sampling minimum dari alat yang anda gunakan?

## 10. Tambahan

Berikan saran atau komentar guna pengembangan lebih lanjut praktikum ini.