SonixDAQ Software (SDK)

Contents

1. Introduction (*Introdução*)

As mentioned before the DAQ is only a receive module and has no control over the transmit. The DAQ SDK uses Texo SDK in order to provide full control over the transmit. This way the transmit sequence can be customized using the Texo and the corresponding pre-beamformed channel data can be received using the DAQ.

Como mencionado anteriormente, o DAQ é um módulo de somente recepção e não há controle sobre a transmissão. O DAQ SDK usa o Texo SDK para fornecer total controle sobre a transmissão. Desta forma, a sequência de transmissão pode ser customizada usando o Texo e os dados de canal pre-beamformed correspondentes podem ser recebidos utilizando o DAQ.

2. Console Program (*Programa Console*)

2.1. Initialization (*Inicialização*)

The DAQ SDK is very similar to the Texo SDK. First, both DAQ and Texo need to be initialized. For proper initialization of the Texo refer to Texo SDK documentation.

O DAQ SDK é muito similar ao Texo SDK. Primeiramente, tanto DAQ quanto Texo precisam ser inicializados. Para uma inicialização adequada do Texo, consulte a documentação do Texo SDK.

The following code initializes the DAQ. The code first checks to see if the DAQ is connected or not. If connected, the software checks to see if the DAQ has already been programmed or not. If programmed, the software will skip the initialization. Otherwise it will start the programming.

O código abaixo inicializa o DAQ. Primeiramente, o código checa se o DAQ está conectado ou não. Se conectado, o software checa se o DAQ já está programado ou não. Se programado, o software irá pular a inicialização. Caso contrário, vai iniciar a programação.

For proper initialization of the DAQ make sure **DAQ_FIRMWARE_PATH** is referring to the correct path for the DAQ firmware

Para uma inicialização adequada do DAQ, tenha certeza de que **DAQ_FIRMWARE_PATH** *é referente ao caminho correto do firmware do DAQ.*

#define DAQ_FIRMWARE_PATH "D:\\mydaq\\fw\\"

Note: Inside the *fw* folder there are different folders for different versions of the DAQ. The software will automatically use the correct folder. Thus, sub folders should not be included in the firmware path.

Nota: Dentro da pasta fw há diferentes pastas para diferentes versões do DAQ. O software irá usar a pasta correta automaticamente. Assim, subpastas não devem ser incluídas no caminho do firmware.

The variable **sampling_80MHz** tells the DAQ either to load the 80MHz sampling bitstream or original 40MHz sampling bitstream. To see this better, if you check inside the firmware folder, you can see that there are two RX bitstreams. One is called *daqrx.bit* and the other is called *daqrx80.bit*.

A variável sampling_80MHz diz ao DAQ para carregar a amostragem de 80MHz ou a amostragem original de 40MHz. Para enxergar isso melhor, se você conferir dentro da pasta do firmware, você pode ver que existem duas RX bitstream. Uma chamanda "daqrx.bi" e outra chamada "daqrx80.bt".

- If *sampling_80MHz = true*, the DAQ will load the 80MHz sampling bitstream.
- If *sampling_80MHz = false*, the DAQ will load the 40MHz sampling bitstream.

- *Se sampling_80MHz = true, o DAQ irá carregar a amaostragem de 80MHz.*
- *Se sampling_80MHz = false, o DAQ irá carregar a amaostragem de 40MHz.*

```
daq = sonixdaq::getInstance();
if (!daq->isConnected())
{
    printf("DAQ not connected or off\n");
    scanf("%c", &sel);
    return -1;
}
else
{
    printf("is connected ... ");
}

if (!daq->isInitialized())
{
    daq->setFirmwarePath(DAQ_FIRMWARE_PATH);
    printf("Programming ... ");
    if (!daq->init(sampling_80MHz))
    {
        printf(daq->getLastError().c_str());
        scanf("%c", &sel);
        return -1;
    }
    printf("Done\n\n");
}
```

The following two steps will finalize the initialization:

Os dois passos seguintes finalizam a inicialização:

```
tex.setSyncSignals(0, 1, 3);
```

This line sets up the synchronization pulses for BNC cables. The inputs for *setSyncSignals* are as follows:

Esta linha configura a os pulsos de sincronização para cabos BNC. Os argumentos para setSyncSignals são os seguintes:

tex.setSyncSignals(input trigger, 1st output trigger, 2nd output trigger)

The above setting sets the Sonix to send line Trigger (i.e. 1) to the first BNC output and 40MHz clock (i.e. 3) to the second BNC output. For further information refer to Texo SDK.

A configuração acima define a Sonix enviar um Trigger de linha (1) para a primeira saída BNC e um clock de 40MHZ (3) para a segunda saída BNC. Para mais informações consulte o Texo SDK.

Note: Clock output is only available on the SonixTouch and Upgraded SonixRP systems.

Nota: A saída clock só está disponível para os sistemas SonixTouch e SonixRP Upgraded.

Note: Setting the output trigger to 2 will activate the frame trigger instead of line trigger. For further information check the Texo SDK.

Nota: Configurando a saída trigger para 2 irá ativar o trigger de frame no lugar do trigger de linha. Para mais informações consulte o Texo SDK.

The last step in the initialization is to open up the DAQ connector. Assuming that the DAQ is connected to the lower probe connector i.e. 3, the following code will open the 3rd connector so that it can listen to the same echoes that are coming back from the transducer simultaneously

O último passo da inicialização é a abertura do conector DAQ. Assumindo que o DAQ está conectado no ultimo conector de probe (3), o código abaixo irá abrir o terceiro conector então o DAQ pode adquirir os mesmos ecos que estão vindo do transdutor, simultaneamente.

```
tex.forceConnector(3);
```

3. Loading a Sequence (*Carregando uma sequência*)

After initialization, a probe needs to be selected and a transmit sequence needs to be loaded into the hardware. These steps are very similar to programming Texo SDK. Some custom transmit sequences are provided as examples. These include: fixed plane wave transmit and transmits with sliding aperture.

Após a inicialização, a probe precisa ser selecionada e uma sequência de transmissão precisa ser carregada no hardware. Estes passos são muito parecidos com programar com o Texo SDK. Algumas sequências de transmissão customizadas são fornecidas como exemplos. Isso inclui: transmissão de onda plana fixa e transmissão com uma abertura "deslizante".

3.1. Running The Sequence for Data Collection (*Rodando a sequência para aquisição de dados*)

Once you trigger run function, both Texo and DAQ will start the data collection simultaneously. For each transmit, Texo will collect one beamformed scan line and DAQ will collect pre-beamformed channel data corresponding to the same transmit.

Depois de ativar a função run, ambos Texo e DAQ irão começar a aquisição de dados simultaneamente. Para cada transmissão, Texo irá coletar uma linha com beamform e o DAQ irá coletar um canal de dados pre-beamformed correspondente à mesma transmissão.

3.2. Saving Data (*Salvando dados*)

Following data collection you can save the data. Once you enter the folder name, the DAQ data will be recorded in the following address:

Seguindo a aquisiçaõ de dados você pode salvar dados. Depois de informar o nome da pasta, os dados do DAQ serão gravados no seguinte endereço:

D:\\DAQDATA\\myfolder

The corresponding texo data will also be recorded in the same folder with the name **data.txo**.

Os dados Texo correspondentes também serão gravados na mesma pasta com o nome data.txo.

4. DAQ Parameters (*Parâmetros do DAQ*)

The followings are the DAQ parameters that can be programmed in the DAQ SDK. The parameter corresponds the same settings that are provided in the DAQ control software

Os itens abaixo são os parâmetros do DAQ que podem ser programados com o DAQ SDK. O parâmetro corresponde a mesma configuração que é fornecida no DAQ control software:

```
	rlprms.lineDuration = 110; // line duration in micro seconds
	rlprms.numSamples = 4000 / (1 + rlprms.decimation); // assuming
3000 samples at 40MHz

	seqprms.freeRun = false;
	seqprms.hpfBypass = false;
	seqprms.divisor = 10; // data size = 16GB / 2^divisor
	seqprms.externalTrigger = true;
	seqprms.externalClock = false; // set to true if external clock is
provided
	seqprms.lnaGain = 1; // 16dB, 18dB, 21dB
	seqprms.pgaGain = 1; // 21dB, 24dB, 27dB, 30dB
	seqprms.biasCurrent = 1; // 0,1,2,...,7
	...
```

4.1. Acquisition Parameters (*Parâmetros de aquisição*)

- lineDuration: acquisition time, this value is in micro seconds
- numSamples: number of samples to be acquired from each channel
- freeRun: sets the DAQ to acquire continuously or stop after defined acquisition data size.
- hpfBypass: by pass the digital high pass filter or not.
- divisor: the size of data to be acquired i.e. dataSize = 16GB / 2^divisor
- externalTrigger: defines whether the DAQ should listen to the external sync signal or not. To have synchronized data, this parameter has to be true. Otherwise, the acquisition of the DAQ will not be in sync with transmit.
- externalClock: defines whether the DAQ should use the external clock or use its own internal clock. Set this to false if external clock is not provided.

- *lineDuration: tempo de aquisição, este valor é dado em microsegundos.*
- *numSamples: número de amostras a serem adquiridas por cada canal.*
- *freeRun: configura o DAQ a adquirir continuamente ou para depois de um dado tamanho de dados.*
- *hpfBypass: filtro digital passa alta.*
- *divisor: tamanho dos dados a serem adquiridados, isto é, dataSize = 16GB/2^divisor*
- *externalTrigger: define se o DAQ deve "ouvir" um sinal sync externo ou não. Para ter dados sincronizados este parametron deve ser "true". Caso contrário, a aaquisição do DAQ não será sincronizada com a trabsmissão.*
- *externalClock: define se o DAQ deve usar um "clock" externo ou usar seu próprio "clock" interno. Configure isso como "false" se um "clock" externo não for fornecido.*

4.2. Gain Parameters (*Parâmetros de Ganho*)

- lnaGain: LNA gain [0:1:2] corresponds to [16dB, 18dB, 21dB].
- pgaGain: PGA gain [0:1:3] corresponds to [21dB, 24dB, 27dB, 30dB].
- biasCurrent: switch gain [0:1:7] where 0 completely turns off the switch.

- *lnaGain: Ganho LNA [0:1:2] corresponde a [16dB, 18dB, 21dB].*
- *pgaGain: Ganho PGA [0:1:3] corresponde a [21dB, 24dB, 27dB, 30dB].*
- *biasCurrent: troca o ganho [0:1:7] onde 0 desativa completamente.*

4.3. TGC Parameters (*Parâmetros do TGC*)

- fixedTGC: defines whether the DAQ should use flat TGC or adjustable TGC.
- fixedTGCLevel: [0:1:100] fixed TGC value. This value is only used if fixedTGC parameter is set to be true.

- *fixedTGC: define se o DAQ deve usar o TGC "flat" ou ajustável.*
- *fixedTGCLevel: [0:1:100] valores de TGC fixos. Estes valores só devem ser usados se o parâmetro fixedTGC for ativado.*

- TGCcurve: If fixedTGC is set to be false, a TGC curve needs to be defined. Currently, this curve is defines by 3 points with X, Y values ranging from 0 to 1 where 1 corresponds to maximum value of TGC for Y and maximum acquisition depth for X.

TGCcurve: se fixedTGC for configurado como "false", uma curva de TGC precisa ser definida. Atualmente, esta curva é definida por 3 pontos com valores (X, Y) em uma extensão de 0 até 1, em que 1 corresponde ao máximo valor de TGC para Y e máxima profundidade de aquisição para X.

```
if (seqprms.fixedTGC)
{
    seqprms.fixedTGCLevel = 100;
}
else
{
    // set TGC curve
    tgc.setX(0, 0.0f);
    tgc.setX(1, 0.5f);
    tgc.setX(2, 0.8f);

    tgc.setY(0, 1.0f);
    tgc.setY(1, 1.0f);
    tgc.setY(2, 1.0f);
}
```

4.4. Sampling Frequency Parameters (*Parâmetros da frequência de amostragem*)

The following two parameters control the sampling frequency of the DAQ data: sampling, decimation. Setting the boolean variable parameters **sampling_80MHz** will take care of sampling frequency in the code as follows:

Os dois parâmetros abaixo controlam a frequência de amostragem dos dados DAQ: amostragem. Dezimação. Definir o parâmetro sampling_80MHz cuidará da frequência de amostragem no código da seguinte forma

```
// sampling and decimation
if (sampling_80MHz)
{
    rlprms.sampling = 80;   // DAQ sampling frequency 80 -> 80
[MHz]
    rlprms.decimation = 0;  // no decimation for 80MHz sampling
}
else
{
```

```
                rlprms.sampling = 40;   // DAQ sampling frequency 40 -> 40
[MHz]
                rlprms.decimation = 0;  // Fs = sampling / (1+decimation)
e.g. decimation = 1 -> Fs=20 MHz
            }
```