GoToDevicePosition FUNCTION

TIMING AND ALTERNATIVE CALLS
Overview

- There are multiple methods of driving a MEMS device from one point to another with individual software commands. These are different commands available in Mirrorcle Software Suite’s various SDKs including C++, Matlab, LabView, Python, and Java for Android.

- This kind of software command-by-command control of MEMS Mirror tip and tilt position gives users a flexibility to take various inputs from user or other systems and convey new commands to the mirror in a software loop.

- Depending on the type of command, there may be various timing overhead needed to communicate the command and to execute it on the Controller and ultimately the mirror. The most significant overhead “delay” is generally the result of the software’s and firmware’s safety features which ensure that the MEMS Mirror is not excited with a step function which would cause overshoot, ringing, and potentially device damage.
Three Methods

- This application notes presents the three different options to the user, how these commands are used and the time for the command to be executed:
  - GoToDevicePosition: ~145ms → Slower, Safest
  - SendDataStream: ~50ms → Faster, potential hazard
  - OutputOffset: ~1ms → Fastest, potential hazard
GoToDevicePosition: ~145ms

- This function gets the device’s current position from the controller and the new position from user inputs, and interpolates a path between the two points.
- The interpolation to create a “smooth” movement between the two points is the main reason for the longer time to process this command.
- The newly calculated path of points are checked to be within bounds, and then downloaded to the controller using SendDataStream command.
- This command is primarily for moving from one point to another in a smooth motion, and not for instantaneous step movements between points.
- GoToDevicePosition supports parameter mSec = 0 which does not interpolate the path. This is equivalent to SendDataStream in delay (~50ms).

```c
float duration;
float sum = 0;
mti->ResetDevicePosition();
for (float x=0; x<1; x+=0.01) {
    clock_t begin = clock();
    mti->GoToDevicePosition( x, x, 0, 1 );
    clock_t end = clock();
    duration = float(end-begin)/CLOCKS_PER_SEC;
    sum+=duration;
}
sum /= 100;
printf("Duration GTDP: %f ms\n", sum*1000);
```
SendDataStream: ~50ms

- This is the main command used in Mirrorcle’s APIs to download an array of points to the MEMS controller. This command can also be used to update the MEMS position one point at a time.
- User provides normalized X, Y positions and an 8-bit Digital output value, which update on next sample. (Fast SampleRate, e.g. 50000 is necessary to minimize delay to execution)
- Note that this command will result in mirror movement only if Controller is already ‘running’, and completed previous already-running frame. Therefore we always start this loop with a ResetDevicePosition command.
- This command results in safe mirror movement only if HardwareFilterBw is safely set.

```c
float duration;
float sum = 0;
mni->ResetDevicePosition();
for (float x=0; x<1; x+=0.01) {
    float data[1];
    unsigned char m[1];
    data[0] = x;
    m[0] = 255;
    clock_t begin = clock();
    mni->SendDataStream( data, data, m, 1);
    clock_t end = clock();
    duration = (float)(end-begin)/CLOCKS_PER_SEC;
    sum+=duration;
}
sum /= 100;
printf("Duration SDS: %f ms\n", sum*1000);
```

Example Code
OutputOffsets: \(~1\text{ms}\)

- Applying new OutputOffsets is the fastest method of moving the MEMS device’s position from individual software calls on a command-by-command basis.
- OutputOffsets is the fastest type of API command as it does not require a reply from the Controller to the PC to confirm.
- This command takes an existing waveform, e.g. MEMS position at 0,0 (the origin), and OutputOffsets applied to offset its position to the new location.
- Note that this command will result in mirror movement only if Controller is already ‘running’, and completed previous already-running frame. Therefore we always start this loop with a ResetDevicePosition command.
- This command results in safe mirror movement only if HardwareFilterBw is safely set.

```c
float duration;
float sum = 0;

mti->ResetDevicePosition();
for (float x=0; x<1; x+=0.0001) {
  clock_t begin = clock();
  mti->SetDeviceParam(MTIParam::OutputOffsets, x, x);
  clock_t end = clock();
  duration = float(end-begin)/CLOCKS_PER_SEC;
  sum+=duration;
}
sum /= 10000;
printf("Duration OO: %f ms\n", sum*1000);
```

Example Code
Thank You for Choosing

Additional Resources:

- Mirrorcle MEMS Mirrors – Technical Overview
- Mirrorcle Documentation Portal
- Mirrorcle Web Page – Support
- Mirrorcle Web Page – Application Notes
- Mirrorcle Web Page – Publications

If you have any further questions or suggestions please email us:
support@mirrorcletech.com