Technical Note

How to find maximum debug frequency

June 2023

iSYSTEM

A TASKING Company

This Technical Note describes how to find a maximum debug interface frequency with a Python script.

Tool requirements:
✓ isystem.connect Python

Python script example is attached to this document.

The script can be used to identify the maximum debug frequency of a connected setup (embedded target & debugger BlueBox debugger). Limiting debug frequency is not strictly defined and it may vary depending on the embedded target used and its connection to the debugger.

What is limiting frequency?

Limiting frequency is a frequency, where debug interface signals between the embedded target and the BlueBox debugger cannot be correctly sampled respectively valid data read due to signal integrity issues. This can happen due to bad PCB layout of debug lines on the embedded target or due to use of a longer debug cable between the BlueBox and the embedded target. Using debug interface clock above the limiting frequency can result in a debug session operating unstable or not operating at all.

Disclaimer: For critical operations (i.e FLASH programming of critical sectors) we recommend to reduce debug frequency to max 80% of the identified frequency by the script.

Python script example

Download and unzip the script example via the link below:

debugFrequencyFinder.7z	

How does this method work?

To find a stable debug interface frequency, several dummy RAM read and RAM write operations are performed at the start of the RAM area. Bisection algorithm tries to increase or decrease debug frequency in steps, depending on the result of the past executed operations. When the frequency step within the bisection algorithm reaches a certain threshold (1% of the difference between maximum and minimum debug frequency set or manual value set by argument *-t* at script execution) the script applies the last stable debug frequency and prompts you whether you want to save the new configuration or not.

Initial debug frequency set in a workspace must be valid to use this test! If you are unsure what frequency to set, set it to some low value which was tested to work.

Following debug protocols are supported using this method:

- CoreSight SWD
- JTAG (CoreSight, TriCore,...)
- TriCore DAP
- RH850 LPD

Using the Python script

1. Use isystem.connect API for running Python scripts.

2. Run *debugFrequencyFinder.py* with iSYSTEM Python interpreter using arguments below:

```
%ISYSTEM_PYTHON%\python.exe debugFrequencyFinder.py -wPATH_TO_WORKSPACE -lRAM_LOCATION
[-mMIN_DEBUG_FREQUENCY -MMAX_DEBUG_FREQUENCY -tFREQUENCY_THRESHOLD]
```

Argument	Description
-w	Path to workspace
-l	RAM start location in hex
-m	Minimal debug frequency (in kHz) to be tested (optional)
-M	Maximal debug frequency (in kHz) to be tested (optional)
-t	Debug frequency threshold (in kHz) (optional)
-v	Verbose output (optional)
%ISYSTEM_PYTHON%	Path to your iSYSTEM Python interpreter

Debug frequency

Debug frequency threshold which is used during the test is calculated with the following formula:

$$FrequencyThreshold = \frac{MaxDebugFrequency - MinDebugFrequency}{100kHz}$$

In cases where the FREQUENCY_THRESHOLD calculation result is less than 100kHz, 100kHz will be used.

Default values:
MIN_DEBUG_FREQUENCY = 100 kHz
MAX_DEBUG_FREQUENCY = 20000 kHz
FREQUENCY_THRESHOLD = 199 kHz (using above formula)

Example

```
%ISYSTEM_PYTHON%\python.exe debugFrequencyFinder.py -wexample.xjrf -l0x20000000 -m300 -M20000 -t500
```

During the script execution do not manually control winIDEA instance which is being used by the Python script.

To set max debug frequency via winIDEA interface follow this tip: How to configure maximum debug interface frequency for best debug performance?