Masterproef FTI: Elektronica-ICT
Voortgangsverslag

Voornaam, Naam	**Dennis Joosens**	**E-mail:**	**dennis.joosens@student.uantwerpen.be**	**VGV**	**6**

ACADEMIEJAAR 2016/2017
VERSLAG INGEDIEND OP: 07/05/2017

Voornaam, naam promotor(s)				
Theo Debrouwere +32 470 653 615 t.debrouwere@televic.com Walter Daems +32 473 335 155 walter.daems@uantwerpen.be				
Data waarop de rapporten werden ingediend	1. 26/02/2017	2. 12/03/2017	3. 26/03/2017	4. 09/04/2017
	5. 23/04/2017	6. 07/05/2017	7.	8.
	9.	10.	11.	12.

ABSTRACT VAN HET ONDERZOEK
Ontwerpen van een proof of concept videoconferencing systeem met een maximale end-to-end latency van 25 ms gebruik makende van het TI AM5728 EVM ontwikkelingsbord met camera module.

Korte omschrijving van de evolutie van het onderzoek tijdens de betrokken periode, met aanduiding van de reeds bekomen resultaten en een planning voor de verdere uitwerking, welke problemen zijn ondervonden en hun oplossingen (totaal minimum twee pagina's - maximum vijf pagina's):

Week 24/04/2017 – 07/05/2017

De opdracht voor deze twee weken bestond uit het uitvoeren van stress tests op het EVM board terwijl het latency bestand wordt uitgevoerd. Daarnaast was het de bedoeling om de audio over het netwerk te sturen en dit door middel van TCP, UDP en RTP te encapsuleren en bijkomend te encoderen met Opus.

1. Uitvoeren van stress tests

Voor het uitvoeren van de stress tests heb ik gebruik gemaakt van de tools die aanwezig waren op het EVM board namelijk de tool genaamd "stress" en "cpuburn". Algemeen kan ik besluiten dat de stress tests geen of nauwelijks invloed hebben op het uitvoeren van het latency programma.

De stress tool is zowat de meest geavanceerde. Hierbij kan je meerdere CPU's op 100% laten draaien, RAM geheugen naar believen vullen en stress uitvoeren op de storage hardware. Ik heb enkele metingen uitgevoerd, telkens met meer stress op de resources. Hier zal ik twee metingen toelichten.

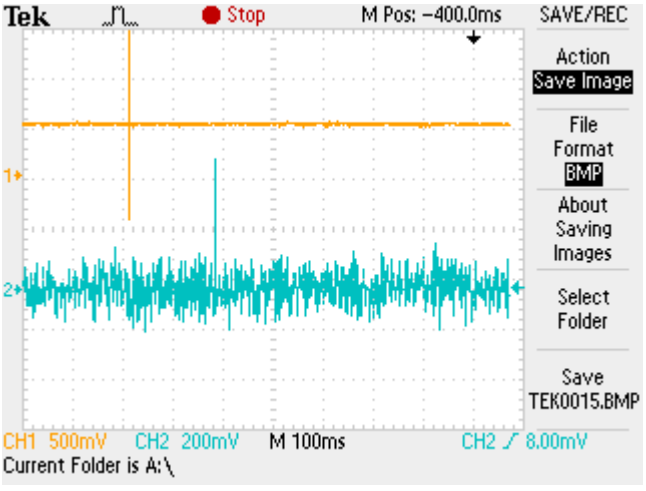
1.1 stress meting

Als ingangssignaal had ik eerst een impuls van 10 kHz gebruikt, ik ben echter overgestapt naar een enkele sinus puls die ik kon triggeren aangezien dit uit ondervinding eenvoudiger op de scoop waar te nemen was. In figuur 1 zie je het resultaat waarbij het oranje signaal input is en het lichtblauwe het output signaal.

Uitvoeren van volgende commando's:

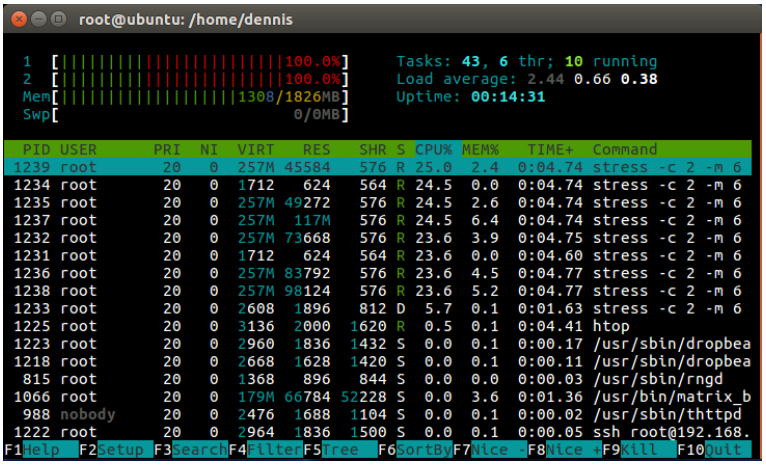
./latency -m 8192 -s 120 (geen polling)

stress -c 2 -m 6 -d 1 (2 cpu's op 100%, RAM vullen met (6 x 256 MB), disk write uitvoeren)



(Fig 1. Latency meting tijdens stress test)

Via het commando htop kon ik het gebruik van de resources in het oog houden:



(Fig 2. htop resource overview)

Op de oscilloscoop meet ik een output latency van 172 ms, in de software 170.67 ms. Alsook voor andere waardes van de buffergrootte bleef dit vrijwel ongewijzigd.

1.2 cpuburn

Als extra test heb ik nog cpuburn gebruikt. Deze tool laat enkel toe om de CPU's te testen. Wat wellicht geen nieuw resultaat zou geven. Na het uitvoeren van het latency.c file met zowel hoge als lage buffer groottes is er geen verandering te merken in latency. Het testen met extra andere tools leek me overbodig aangezien de stress tool je voldoende opties biedt.

2. Audio encapsulatie en encodering

Als eerste test voeren we een simpele ping uit van de VM naar het EVM board. Het ping commando gebruikt het ICMP protocol. Het wordt voornamelijk gebruikt voor simpele query's uit te voeren en error reporting. In figuur 3 worden 15 ECHO_REQUESTS gestuurd als deze een ECHO_REPLY krijgen, dan krijgen we de Round-Trip-Time te zien. Hierbij sturen we 512 bytes als payload. Deze worden aanzien als 520 bytes omwille van een 8 bytes ICMP header. Verder vinden we ook nog 540 bytes terug omdat de ICMP datagrams geëncapsuleerd en verzonden worden via het IP protocol, die een header heeft van 20 bytes (512 + 8 + 20). Uit figuur 3 halen we een gemiddelde RTT van 0,413 ms.

```
root@ubuntu:/home/dennis/network# ping 192.168.10.10 -s 512 -c 15
PING 192.168.10.10 (192.168.10.10) 512(540) bytes of data.
520 bytes from 192.168.10.10: icmp_seq=1 ttl=64 time=0.373 ms
520 bytes from 192.168.10.10: icmp_seq=2 ttl=64 time=0.628 ms
520 bytes from 192.168.10.10: icmp_seq=3 ttl=64 time=0.415 ms
520 bytes from 192.168.10.10: icmp_seq=4 ttl=64 time=0.406 ms
520 bytes from 192.168.10.10: icmp_seq=5 ttl=64 time=0.410 ms
520 bytes from 192.168.10.10: icmp_seq=6 ttl=64 time=0.432 ms
520 bytes from 192.168.10.10: icmp_seq=7 ttl=64 time=0.414 ms
520 bytes from 192.168.10.10: icmp_seq=8 ttl=64 time=0.428 ms
520 bytes from 192.168.10.10: icmp_seq=9 ttl=64 time=0.281 ms
520 bytes from 192.168.10.10: icmp_seq=10 ttl=64 time=0.549 ms
520 bytes from 192.168.10.10: icmp_seq=11 ttl=64 time=0.281 ms
520 bytes from 192.168.10.10: icmp_seq=12 ttl=64 time=0.403 ms
520 bytes from 192.168.10.10: icmp_seq=13 ttl=64 time=0.408 ms
520 bytes from 192.168.10.10: icmp_seq=14 ttl=64 time=0.418 ms
520 bytes from 192.168.10.10: icmp_seq=15 ttl=64 time=0.350 ms

--- 192.168.10.10 ping statistics ---
15 packets transmitted, 15 received, 0% packet loss, time 13999ms
rtt min/avg/max/mdev = 0.281/0.413/0.628/0.084 ms
```

(Fig 3. Uitvoeren van ping commando)

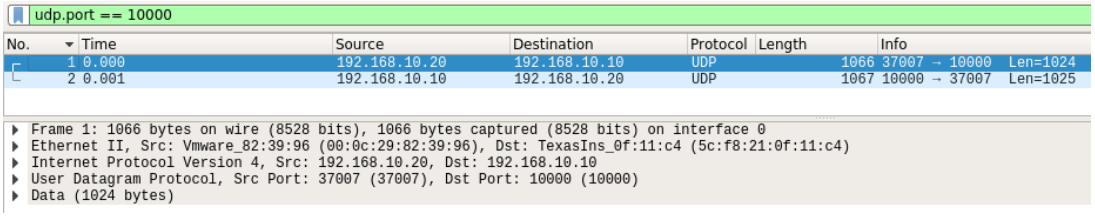
Voor het verzenden van data via TCP en UDP ben ik begonnen met het opzetten van een simpele TCP en UDP client in C om dan random bytes te verzenden. Merk op dat de RTT hier wordt berekend. Dit geeft een grotere latency. De latency houdt in dat de data verzonden wordt naar de server, daar verwerkt wordt en weer teruggestuurd wordt naar de client. In de praktijk zal de latency om van client naar server te communiceren maar de helft of zelfs minder bedragen.

Uit tabel 1 blijkt dat TCP een grotere vertraging geeft. Wat niet abnormaal is aangezien een three-way handshake wordt opgezet, de aangekomen data bij de ontvanger wordt acknowledged en de verbinding wordt beëindigd door beide partijen. Bijkomend zal ook het type header een kleine invloed hebben op de latency (8 bytes versus 20 bytes).

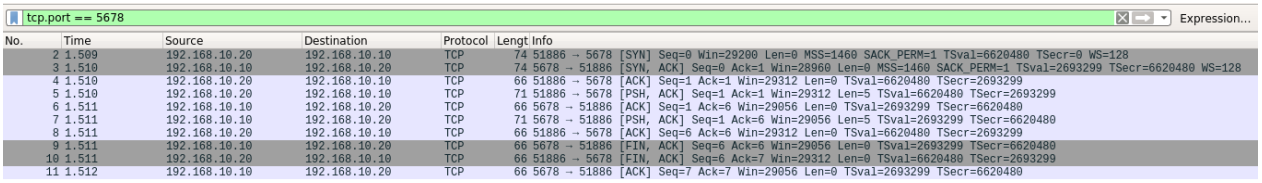
send "hello"				send random 512 bytes				send random 1024 bytes		
	#	UDP RTT (in ms)	TCP RTT (in ms)		#	UDP RTT (in ms)	TCP RTT (in ms)		#	UDP RTT (in ms)	TCP RTT (in ms)
	1	0,332	1,427		1	0,282	1,629		1	0,37	1,135
	2	0,269	1,755		2	0,179	1,258		2	0,312	0,532
	3	0,407	1,181		3	0,157	1,51		3	0,18	0,456
	4	0,302	1,332		4	0,278	1,132		4	0,288	0,54
	5	0,132	0,62		5	0,203	0,413		5	0,251	1,073
	6	0,136	1,023		6	0,238	1,146		6	0,178	0,457
	7	0,144	1,542		7	0,241	0,732		7	0,352	1,185
	8	0,228	0,96		8	0,249	1,433		8	0,251	1,257
	9	0,138	0,503		9	0,303	0,883		9	0,153	1,333
	10	0,254	0,624		10	0,409	0,85		10	0,289	1,685
	11	0,339	0,534		11	0,216	0,529		11	0,522	1,536
	12	0,143	0,653		12	0,292	0,419		12	0,147	1,63
	13	0,219	1,362		13	0,241	0,779		13	0,267	0,83
	14	0,308	1,833		14	0,383	0,777		14	0,22	1,04
	15	0,429	0,859		15	0,498	0,509		15	0,183	1,368
	min	0,132	0,503		**min**	0,157	0,413		**min**	0,147	0,456
	max	0,429	1,833		**max**	0,498	1,629		**max**	0,522	1,685
	avg	**0,252**	**1,080533333**		**avg**	**0,277933333**	**0,933266667**		**avg**	**0,2642**	**1,070466667**
	stdev	0,097061493	0,432751794		**stdev**	0,088133585	0,385864305		**stdev**	0,096039714	0,408328931

(Tabel 1. TCP/UDP Round-Trip-Time)

Als we de UDP data vergelijken met de resultaten van het ping commando, komen deze vrij hard overeen. Beide werken op ongeveer dezelfde manier. Ze gebruiken geen handshaking en ze hebben allebei een 8 byte header.



(Fig 4. UDP datagram echo back in Wireshark)



(Fig 5. TCP packet echo back in Wireshark)

Momenteel ben ik bezig om het latency en UDP stuk samen te voegen. Er wordt data verzonden, maar er loopt toch nog iets verkeerd bij het zenden van de data.

Volgende zaken wil ik nog verwezenlijken in de komende weken

- RTP test setup en latency vergelijken met UDP
- UDP en latency bestand samenvoegen
- TCP en latency bestand samenvoegen
- UDP en TCP latency vergelijken
- Audio encodering met Opus via RTP en/of UDP

Extra informatie
Bijgewoonde seminaries, presentaties, workshops, bedrijfsbezoeken etc in deze periode (onderwerp, datum, korte samenvatting en beoordeling)
1.
2.
3.
4.
Nieuwe contacten gemaakt in deze periode (naam, voornaam, e-mail, telefoonnummer, bedrijf, functie, extra opmerkingen)
1.
2.
3.
4.
Gelezen artikels, boeken, interviews, etc (titel, auteurs, aantal blzn., korte beschrijving, eigen beoordeling (wat is de meerwaarde voor het onderzoek))

http://www.alsa-project.org/main/index.php/Test_latency.c
http://www.alsa-project.org/alsa-doc/alsa-lib/_2test_2latency_8c-example.html
Boek - CCNA Exploration Companion Guide Network Fundamentals - Mark A. Dye, Rick McDonald, Antoon W. Rufi
https://tools.ietf.org/html/rfc3550
https://tools.ietf.org/html/rfc7587
https://tools.ietf.org/html/rfc777
Visie en eventuele commentaar van de promotor