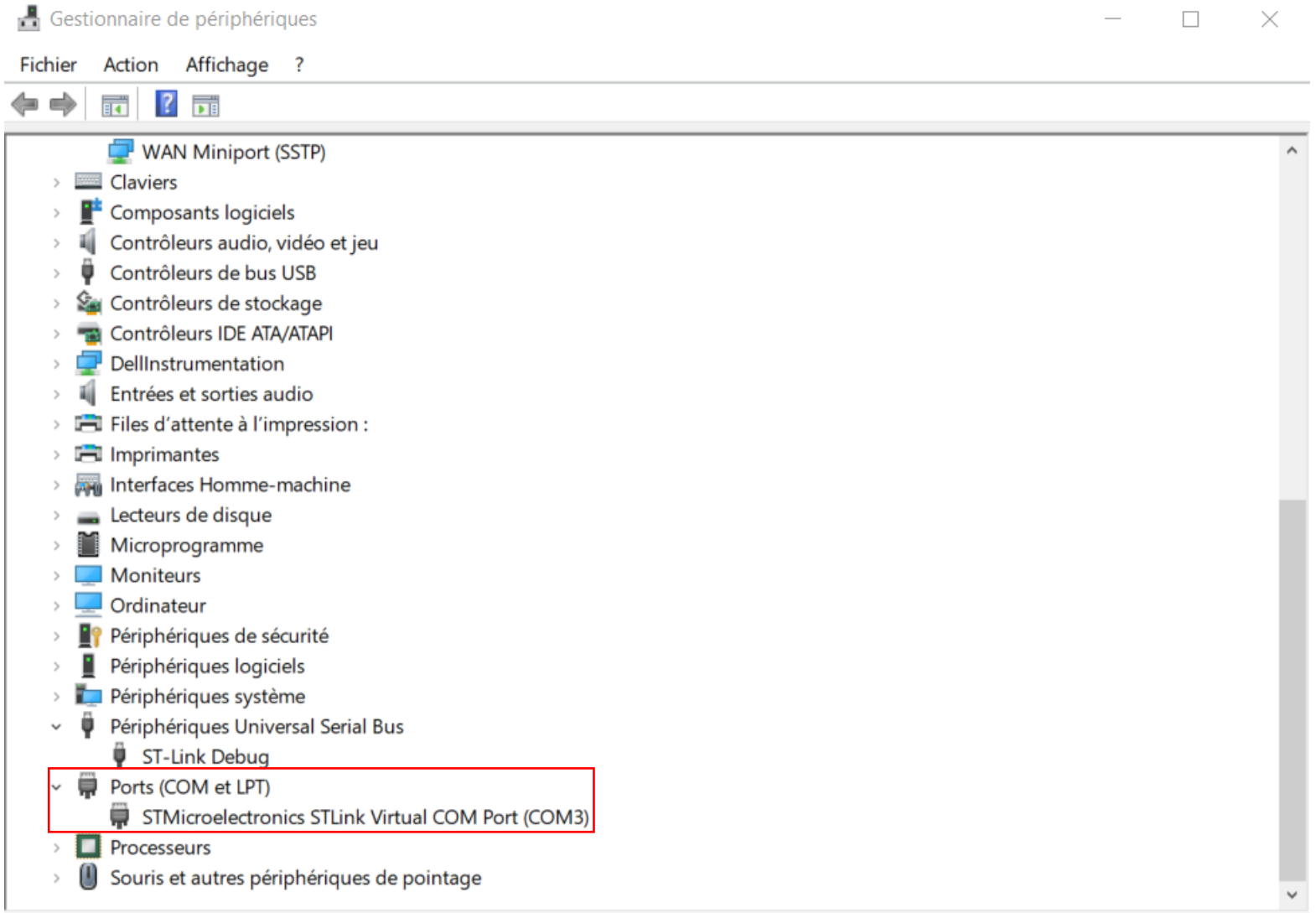
Learning-ADC_TO_UART

Ce document résume mes observations suite à la programmation de la carte de développement : « Nucleo-L476RG ».

L’objectif est d’enregistrer un signal en entrée de l’ADC et lire le buffer en utilisant la liaison UART.



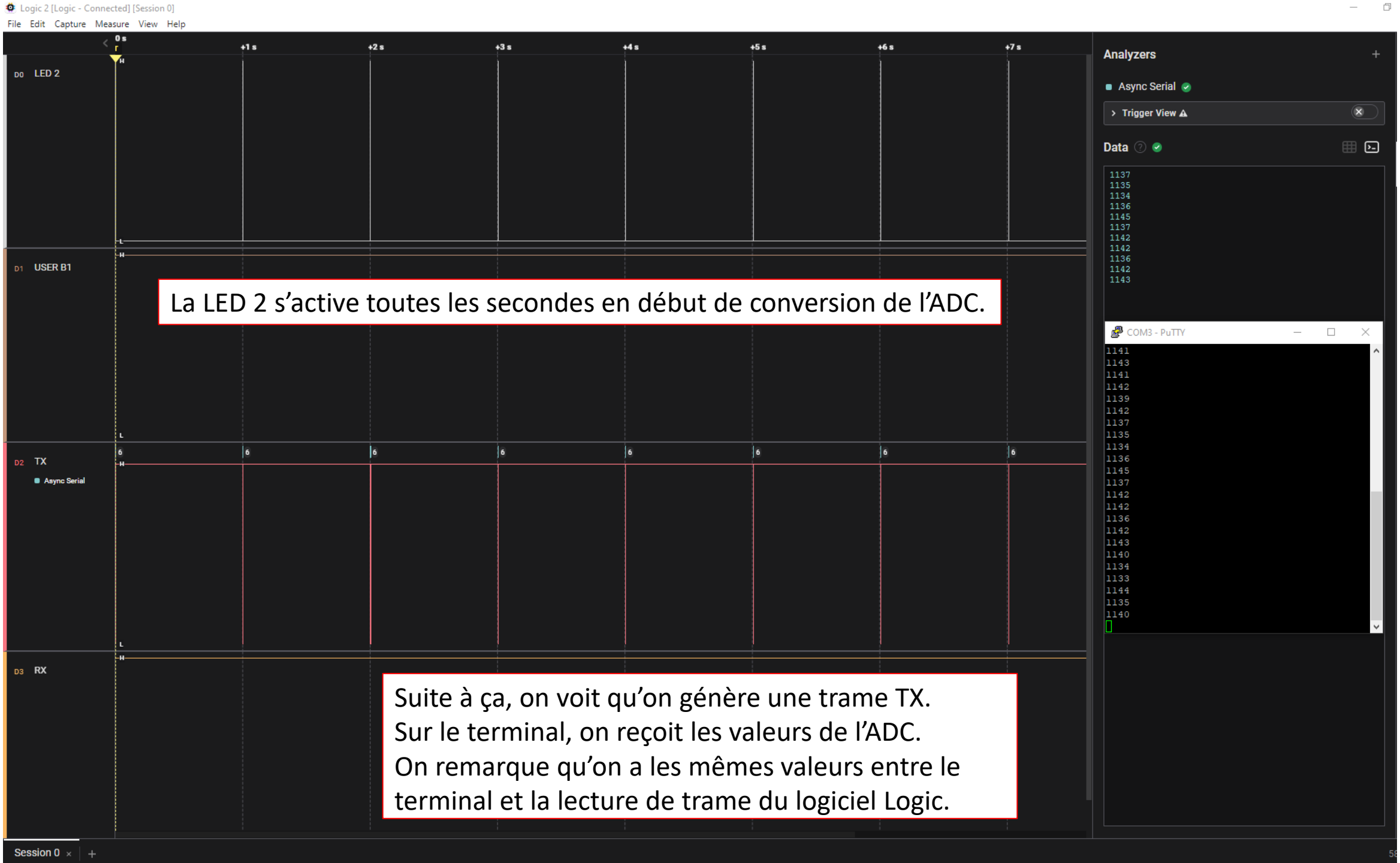
J'ai utilisé la liaison série accessible via le COM3 du STLink.

Première étape

On travail avec un ADC et une liaison UART dans la boucle « While ».

On répète ces actions toutes les secondes :

1) On fait une conversion ADC.
2) On a une valeur sur 12 bits (0 à 4094).
3) On passe cette valeur dans un tableau chiffre par chiffre en « char ».
4) On transmet ce buffer à l'UART.
5) L'UART transmet via TX le buffer.
6) On lit la valeur sur le terminal.


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
0 s
+1 s
+2 s
+3 s
+4 s
+5 s
+6 s
+7 s
D0 LED 2
D1 USER B1
D2 TX
Async Serial
D3 RX
La LED 2 s'active toutes les secondes en début de conversion de l'ADC.
Suite à ça, on voit qu'on génère une trame TX.
Sur le terminal, on reçoit les valeurs de l'ADC.
On remarque qu'on a les mêmes valeurs entre le terminal et la lecture de trame du logiciel Logic.
Analyzers
Async Serial
Trigger View
Data
1137
1135
1134
1136
1145
1137
1142
1142
1136
1142
1143
COM3 - PuTTY
1141
1143
1141
1142
1139
1142
1137
1135
1134
1136
1145
1137
1142
1142
1136
1142
1143
1140
1134
1133
1144
1135
1140
Session 0

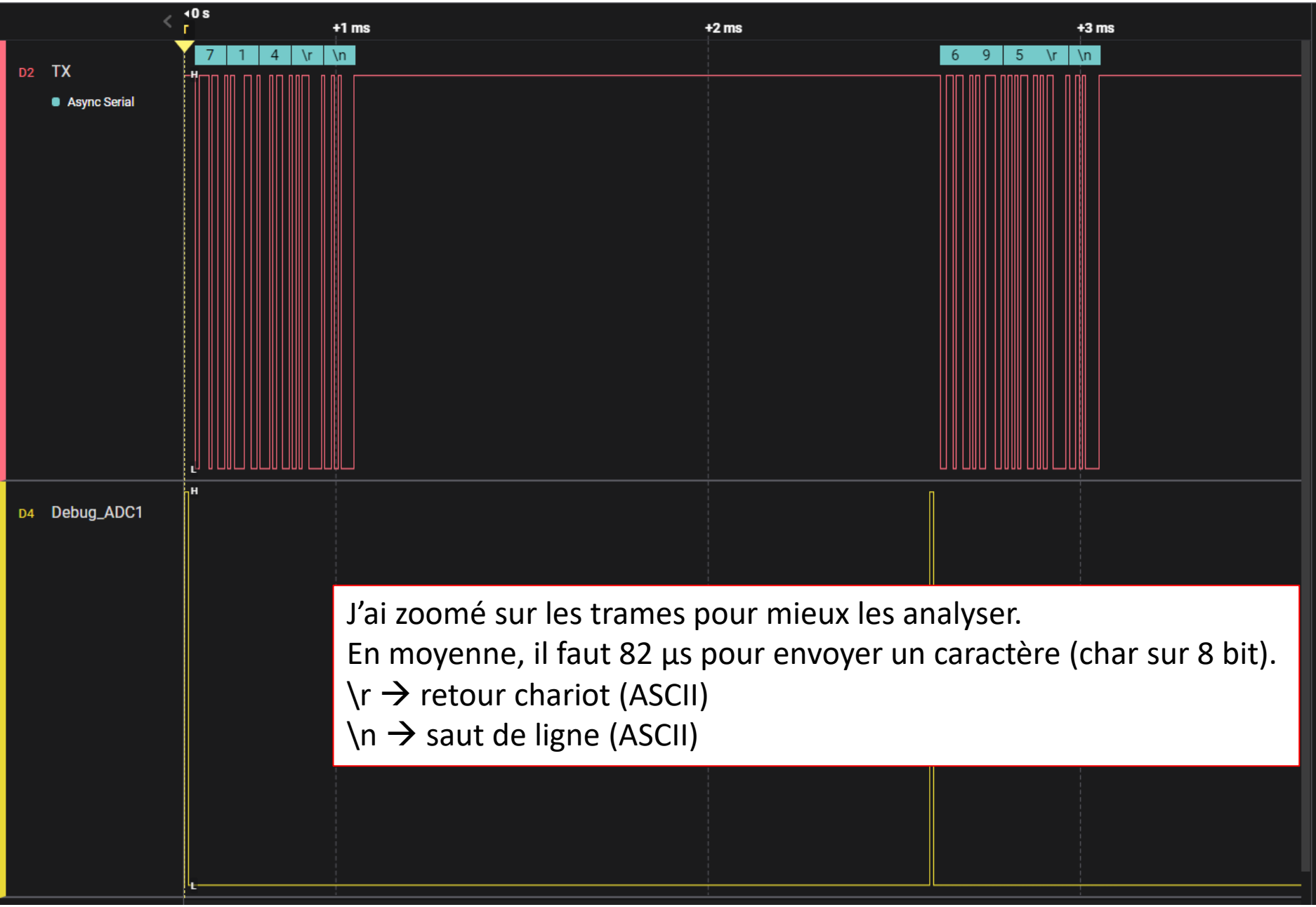

Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
1 s : 1 ms
+0.6 ms
+0.7 ms
+0.8 ms
+0.9 ms
1 s : 2 ms
+0.1 ms
+0.2 ms
+0.3 ms
D0 LED 2
D1 USER B1
D2 TX
Async Serial
D3 RX
0x31
0x31
0x33
0x35
0x0D
0x0A
L'ADC met environ 11 µs pour faire une conversion.
On déchiffre la trame : 0x31 = 1, 0x31 = 1, 0x33 = 3, 0x35 = 5, 0x0D = retour chariot, 0x0A = saut de ligne.
Timing Markers
P1
3
P1
Click "+" to add Timing Markers
Measurements
M0
Δ11 µs
N falling 1
N rising 1
f min N/A
f max N/A
f mean N/A
T std N/A
Notes
Click to add a description of your capture
Session 0


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
+1 s : 1 ms
+0.6 ms
+0.7 ms
+0.8 ms
+0.9 ms
1 s : 2 ms
+0.1 ms
+0.2 ms
+0.3 ms
D0 LED 2
D1 USER B1
D2 TX
Async Serial
D3 RX
0x31
0x31
0x33
0x35
0x0D
0x0A
Le logiciel Logic possède un terminal pour lire les trames voir onglet « Data ». C'est pratique.
Analyzers
Async Serial
Trigger View
Data
1137
1135
1134
1136
1145
1137
1142
1142
1136
1142
1143
Session 0

J’ai supprimé le retour chariot pour voir son impact.

Logic 2 [Logic - Connected] [Session 0]

File Edit Capture Measure View Help



J'ai zoomé sur les trames pour mieux les analyser.
En moyenne, il faut 82 µs pour envoyer un caractère (char sur 8 bit).
\r → retour chariot (ASCII)
\n → saut de ligne (ASCII)

Analyzers

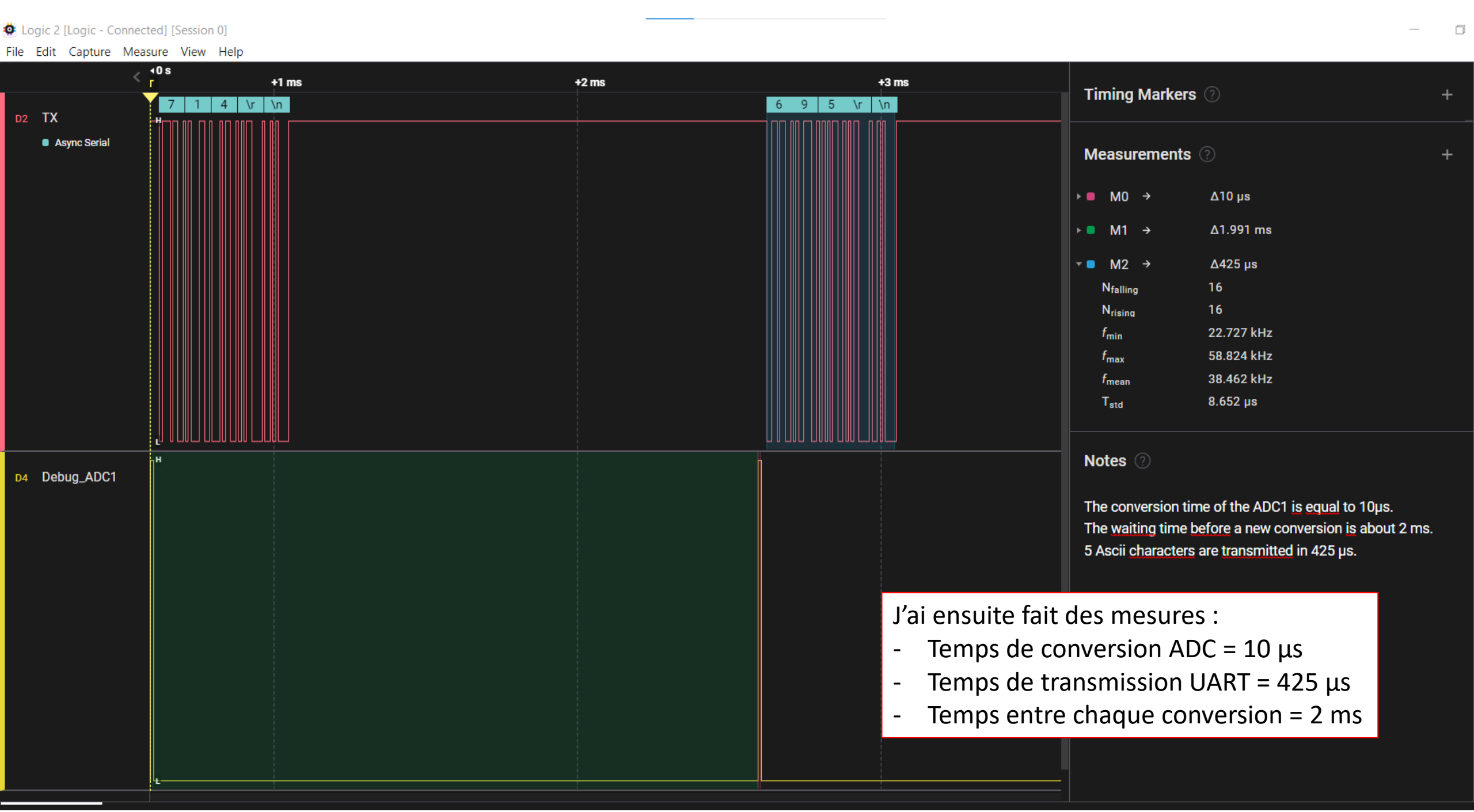
Async Serial

Trigger View

Data

Type to search

Type	Start	Duration	data
data	622 µs	82 µs	7
data	709 µs	82 µs	1
data	796 µs	82 µs	4
data	882 µs	82 µs	\r
data	969 µs	82 µs	\n
data	2.623 ms	82 µs	6
data	2.709 ms	82 µs	9
data	2.796 ms	82 µs	5
data	2.883 ms	82 µs	\r
data	2.97 ms	82 µs	\n



J'ai ensuite fait des mesures :

- Temps de conversion ADC = 10 µs
- Temps de transmission UART = 425 µs
- Temps entre chaque conversion = 2 ms

Première étape (Version 2)

On travail avec un ADC et une liaison UART dans la boucle « While ».

On répète ces actions toutes les secondes :

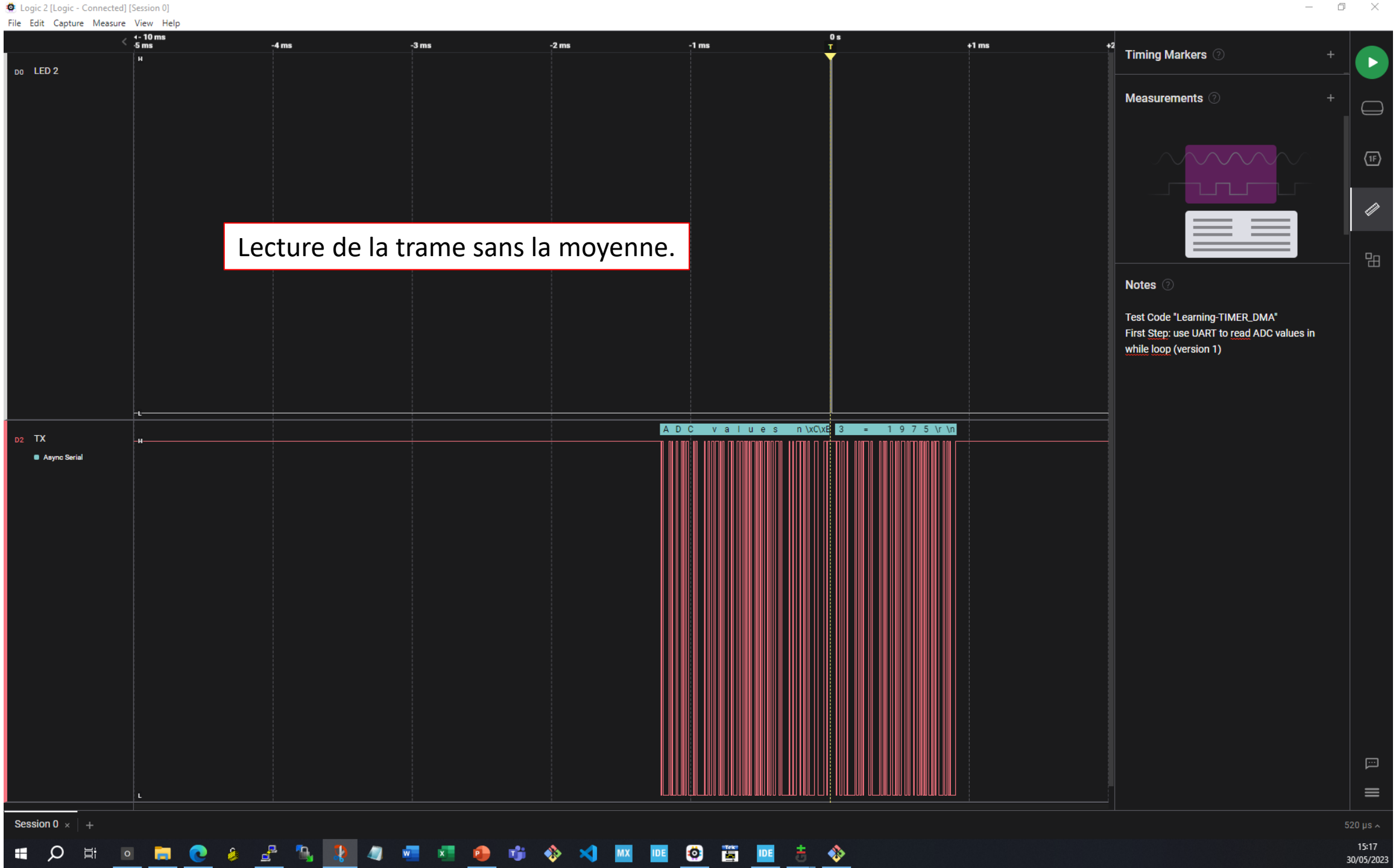
1) On fait une conversion ADC.
2) On a une valeur sur 12 bits (0 à 4094).
3) On passe cette valeur dans un tableau chiffre par chiffre en « char ».
4) On transmet ce buffer à l'UART.
5) L'UART transmet via TX le buffer.
6) On lit la valeur sur le terminal.

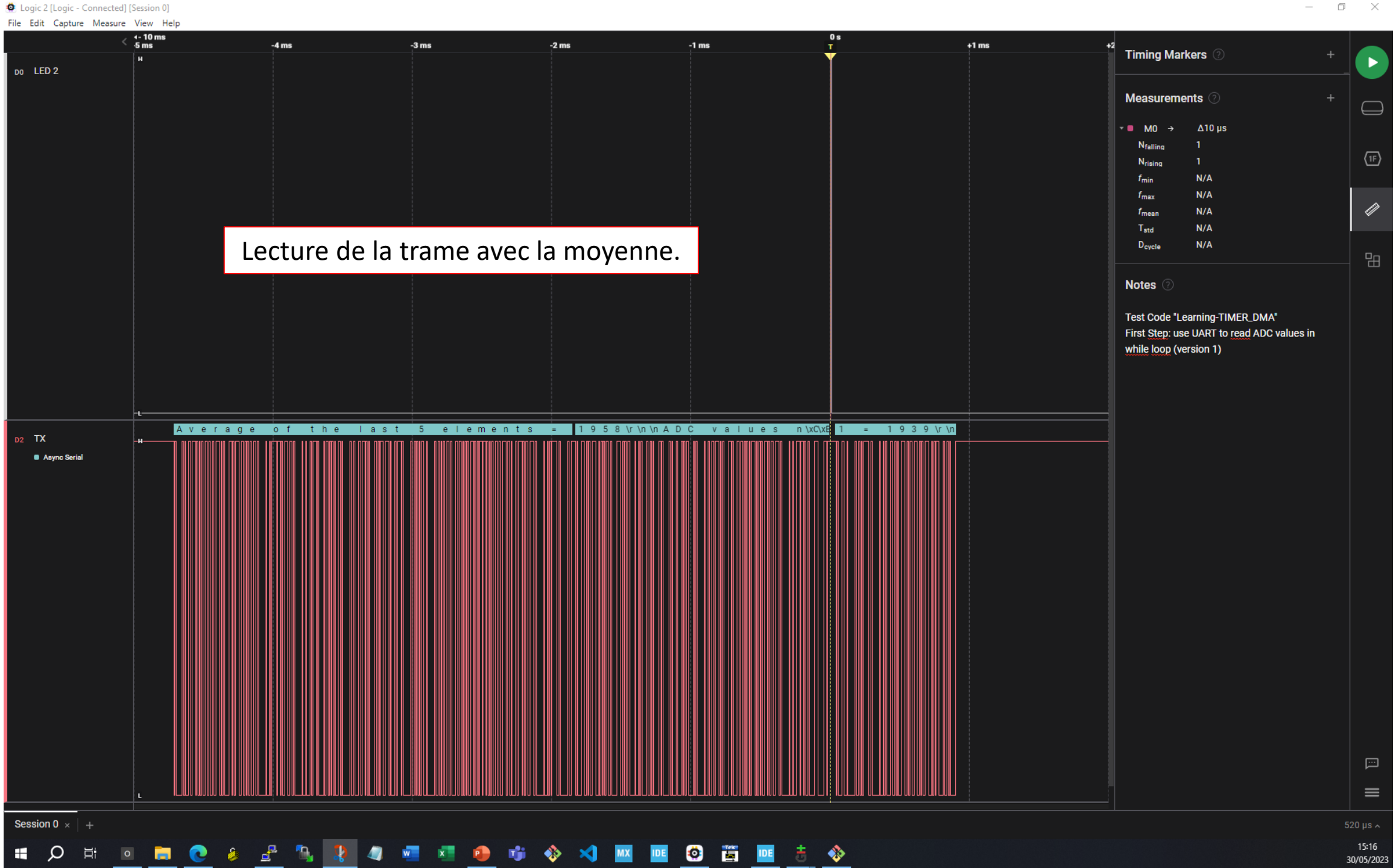
Dans cette version, on stock les 5 dernières valeurs périodiquement pour afficher la moyenne suite à 5 conversion de l'ADC. J'ai ajouté ce bout de code pour avoir une meilleur lisibilité du capteur. J'ai remarqué que les valeurs varient de +/- 20. La moyenne permet de lisser cette variation.

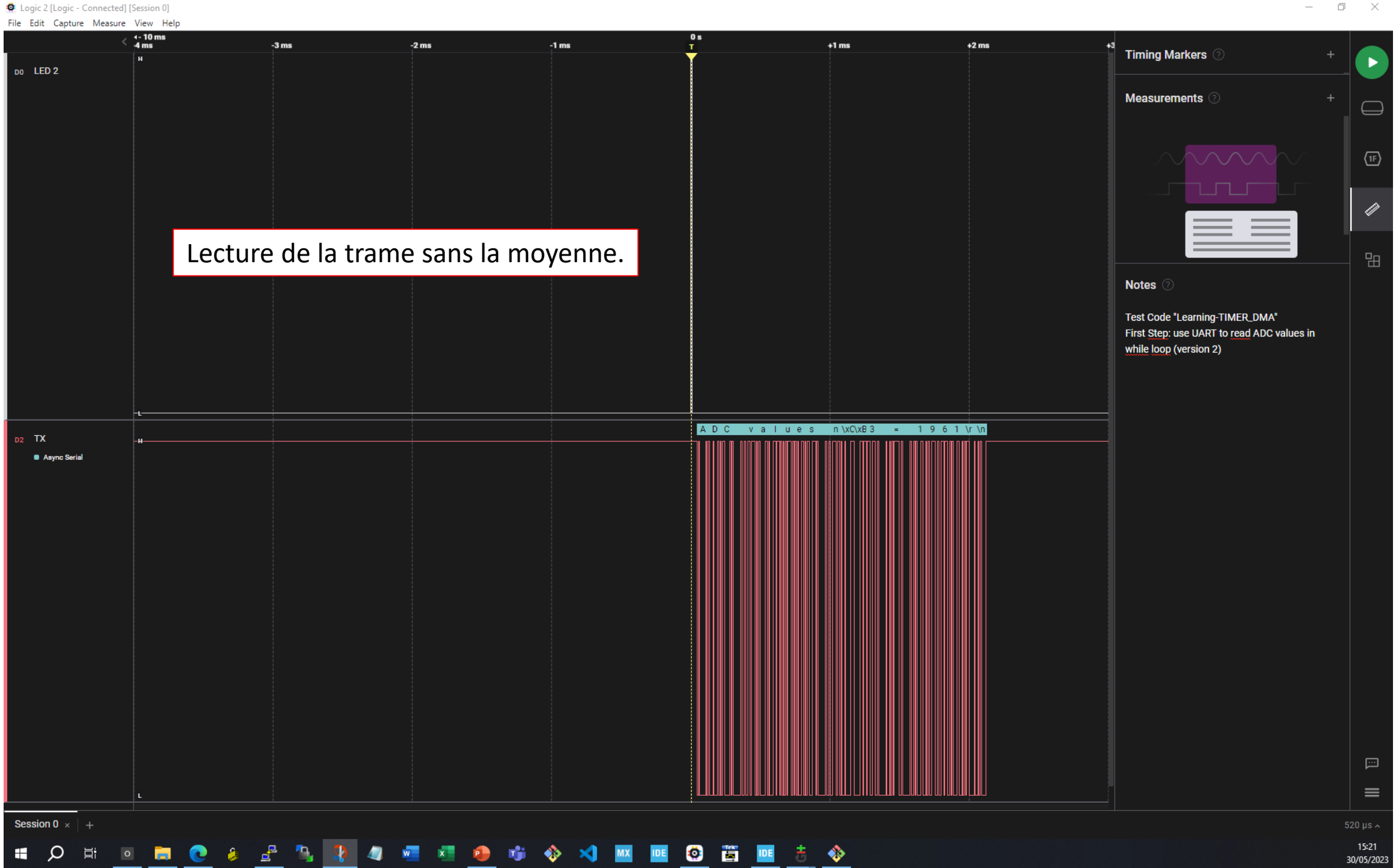
COM3 - PuTTY

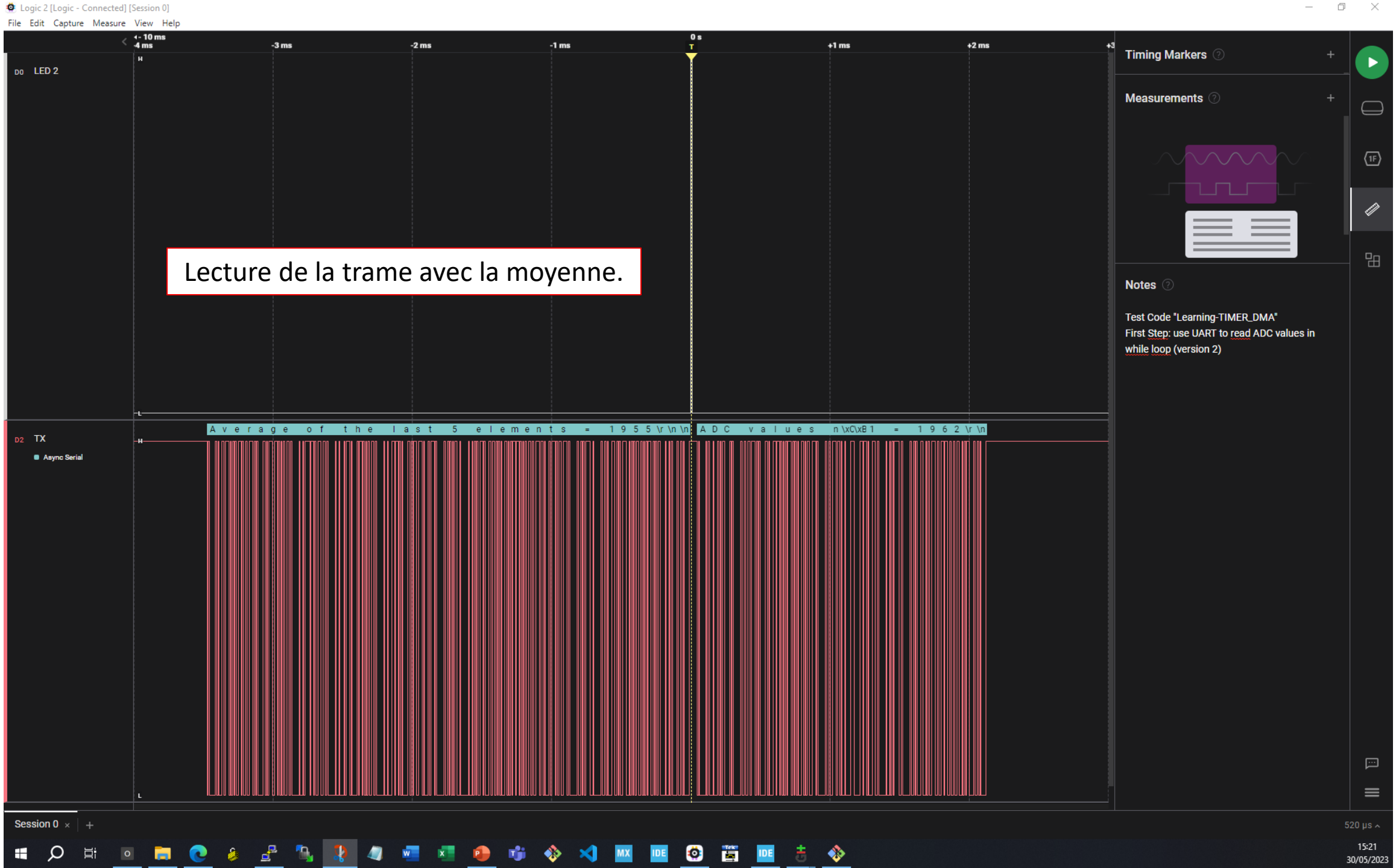
```
ADC values n°2 = 1956
ADC values n°3 = 1926
ADC values n°4 = 1947
ADC values n°5 = 1956
Average of the last 5 elements = 1950

ADC values n°1 = 1939
ADC values n°2 = 1967
ADC values n°3 = 1941
ADC values n°4 = 1975
ADC values n°5 = 1958
Average of the last 5 elements = 1956

ADC values n°1 = 1941
ADC values n°2 = 1926
ADC values n°3 = 1943
ADC values n°4 = 1931
ADC values n°5 = 1934
Average of the last 5 elements = 1935

ADC values n°1 = 1967
ADC values n°2 = 1954
ADC values n°3 = 1960
ADC values n°4 = 1946
ADC values n°5 = 1960
Average of the last 5 elements = 1957

ADC values n°1 = 1956
ADC values n°2 = 1933
ADC values n°3 = 1963
ADC values n°4 = 1955
ADC values n°5 = 1935
Average of the last 5 elements = 1948

ADC values n°1 = 1941
ADC values n°2 = 1966
ADC values n°3 = 1956
ADC values n°4 = 1973
ADC values n°5 = 1960
Average of the last 5 elements = 1959

ADC values n°1 = 1926
ADC values n°2 = 1971
ADC values n°3 = 1947
ADC values n°4 = 1972
ADC values n°5 = 1923
Average of the last 5 elements = 1947
```

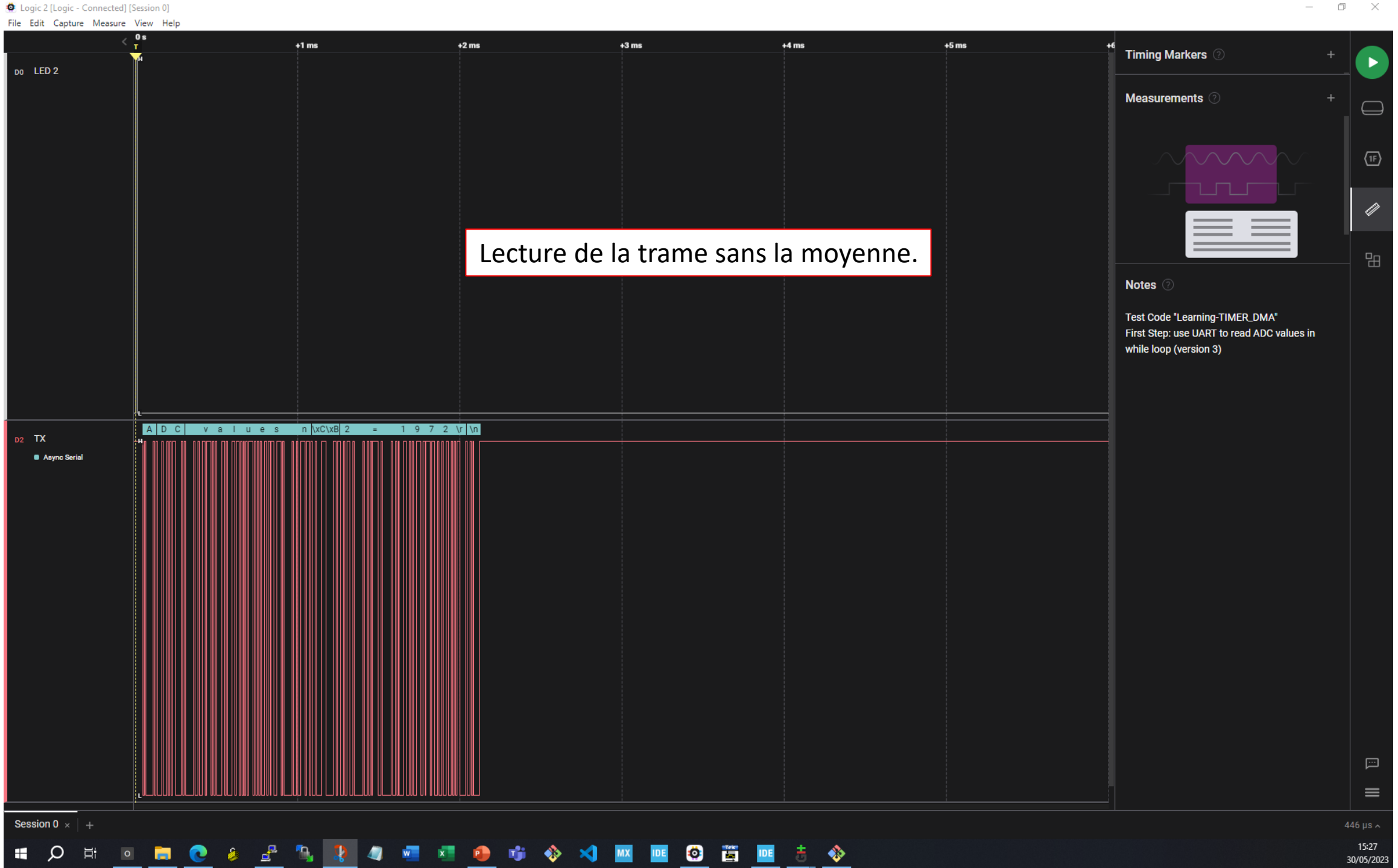
Voici le nouvel affichage.
Toute les 5 valeurs, je fais une moyenne.

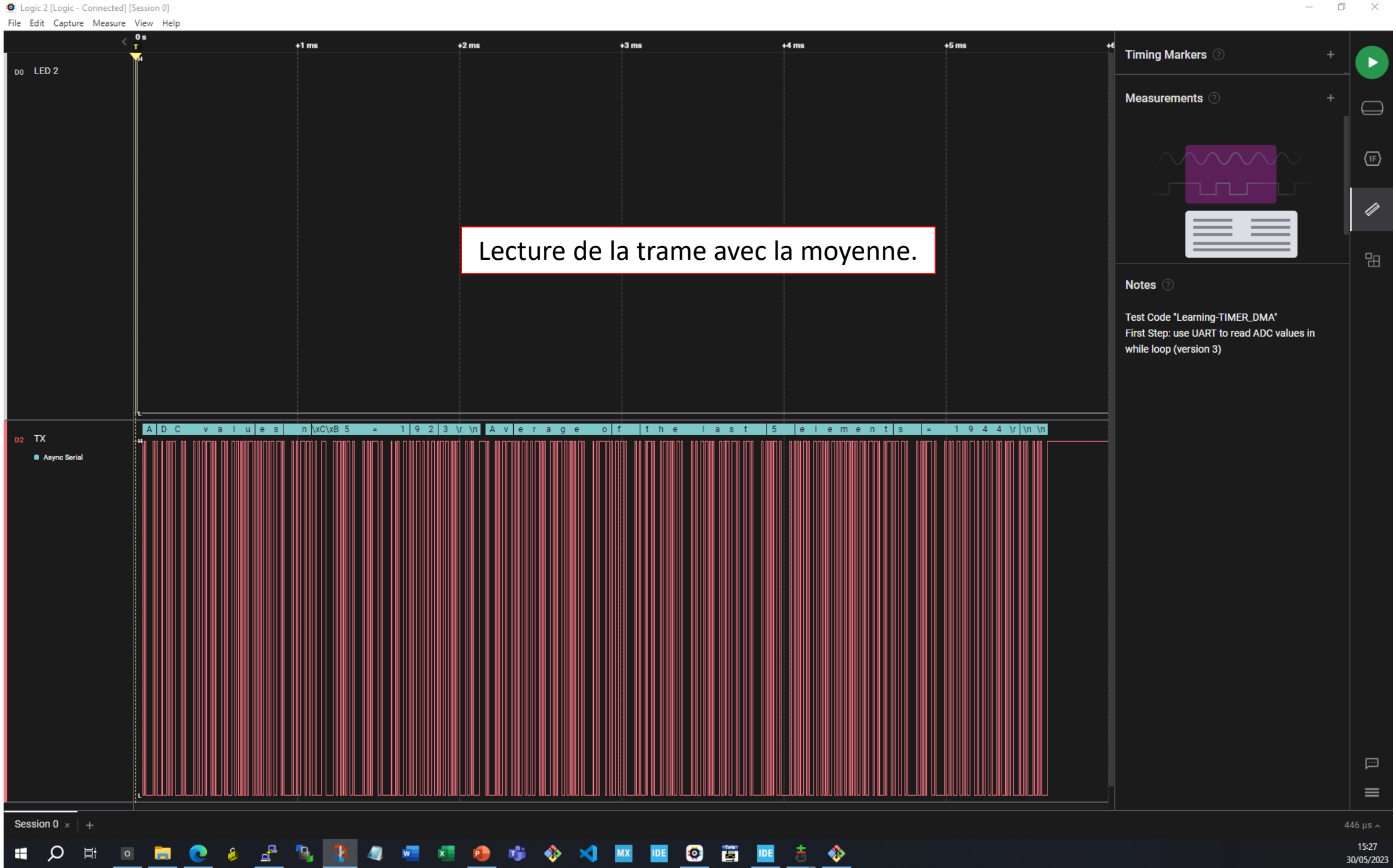

Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
-10 ms
-5 ms
-4 ms
-3 ms
-2 ms
-1 ms
0 s
+1 ms
D0 LED 2
Lecture de la trame sans la moyenne.
D2 TX
Async Serial
A D C v a l u e s n \xC\xB 3 = 1 9 7 5 \r \n
Timing Markers
Measurements
Notes
Test Code "Learning-TIMER_DMA"
First Step: use UART to read ADC values in while loop (version 1)
Session 0
520 µs
15:17
30/05/2023


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
-10 ms
-5 ms
-4 ms
-3 ms
-2 ms
-1 ms
0 s
+1 ms
D0 LED 2
Lecture de la trame avec la moyenne.
D2 TX
Async Serial
Average of the last 5 elements = 1958 \r \n \n ADC values n \xC\xB 1 = 1939 \r \n
Timing Markers
Measurements
M0 → Δ10 µs
N falling 1
N rising 1
f min N/A
f max N/A
f mean N/A
T std N/A
D cycle N/A
Notes
Test Code "Learning-TIMER_DMA"
First Step: use UART to read ADC values in while loop (version 1)
Session 0
520 µs
15:16
30/05/2023


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
-10 ms
-4 ms
-3 ms
-2 ms
-1 ms
0 s
+1 ms
+2 ms
D0 LED 2
Lecture de la trame sans la moyenne.
D2 TX
Async Serial
A D C v a l u e s n \xC\xB 3 = 1 9 6 1 \r \n
Timing Markers
Measurements
Notes
Test Code "Learning-TIMER_DMA"
First Step: use UART to read ADC values in
while loop (version 2)
Session 0
520 µs
15:21
30/05/2023


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
D0 LED 2
D2 TX
Async Serial
Average of the last 5 elements = 1955 \r \n \n
ADC values n \xC\xB1 = 1962 \r \n
Lecture de la trame avec la moyenne.
Timing Markers
Measurements
Notes
Test Code "Learning-TIMER_DMA"
First Step: use UART to read ADC values in while loop (version 2)
Session 0
520 µs
15:21
30/05/2023


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
0 s
+1 ms
+2 ms
+3 ms
+4 ms
+5 ms
D0 LED 2
Lecture de la trame sans la moyenne.
D2 TX
Async Serial
A D C v a l u e s n \xC\xB 2 = 1 9 7 2 \r \n
Timing Markers
Measurements
Notes
Test Code "Learning-TIMER_DMA"
First Step: use UART to read ADC values in while loop (version 3)
Session 0
446 µs
15:27
30/05/2023


Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
0 s
+1 ms
+2 ms
+3 ms
+4 ms
+5 ms
D0 LED 2
Lecture de la trame avec la moyenne.
D2 TX
Async Serial
ADC values n\xC\xB5 = 1923\r\n
Average of the last 5 elements = 1944\r\n\n
Timing Markers
Measurements
Notes
Test Code "Learning-TIMER_DMA"
First Step: use UART to read ADC values in while loop (version 3)
Session 0
446 µs
15:27
30/05/2023

Deuxième étape

J’utilise une DMA sur la transmission de l’UART pour envoyer un buffer avec beaucoup de valeur.

L’objectif de l’utilisation de la DMA dans ce cas est de ne pas monopoliser le CPU pour transmettre un grand buffer.

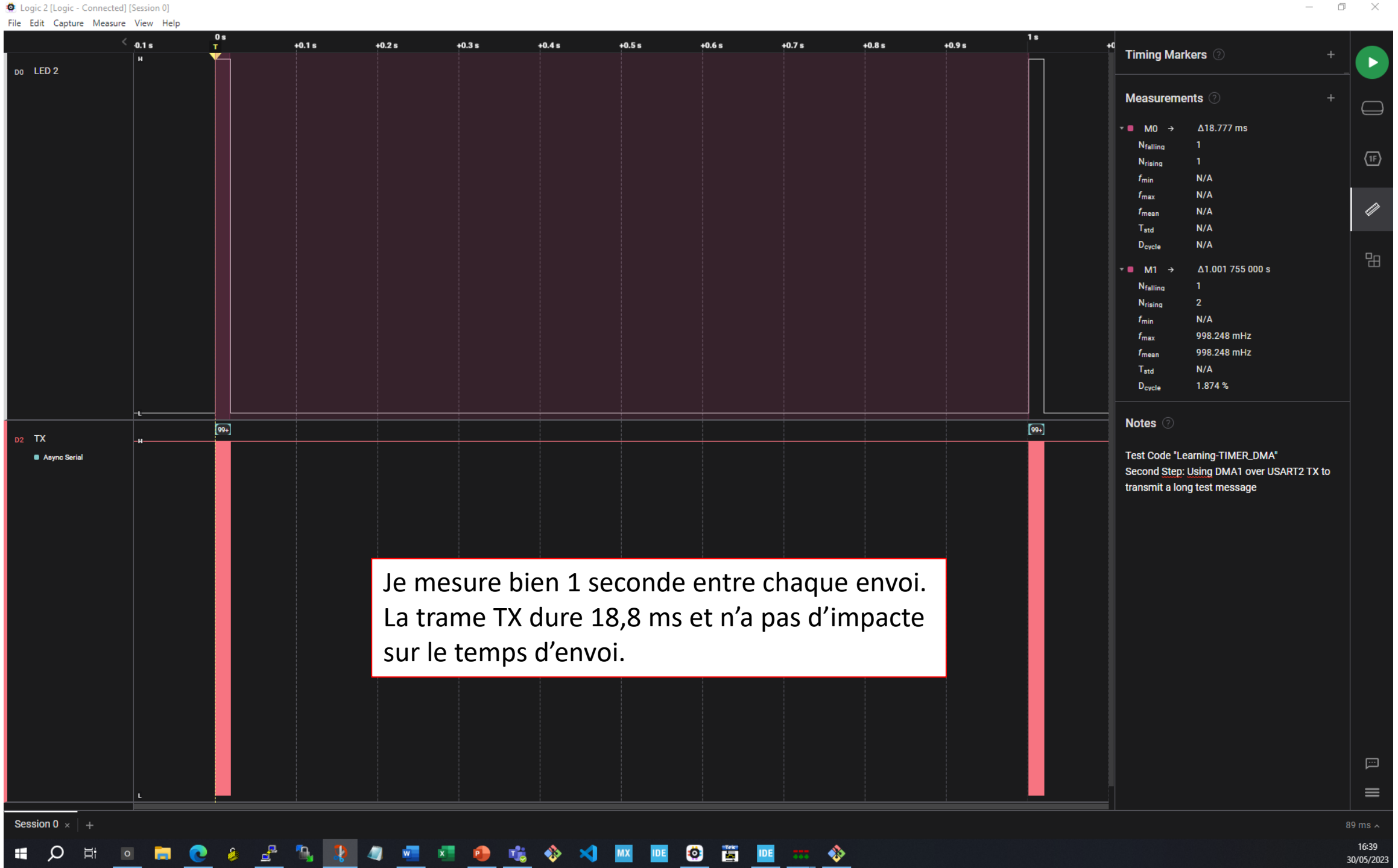
C’est une première utilisation de la DMA pour moi.

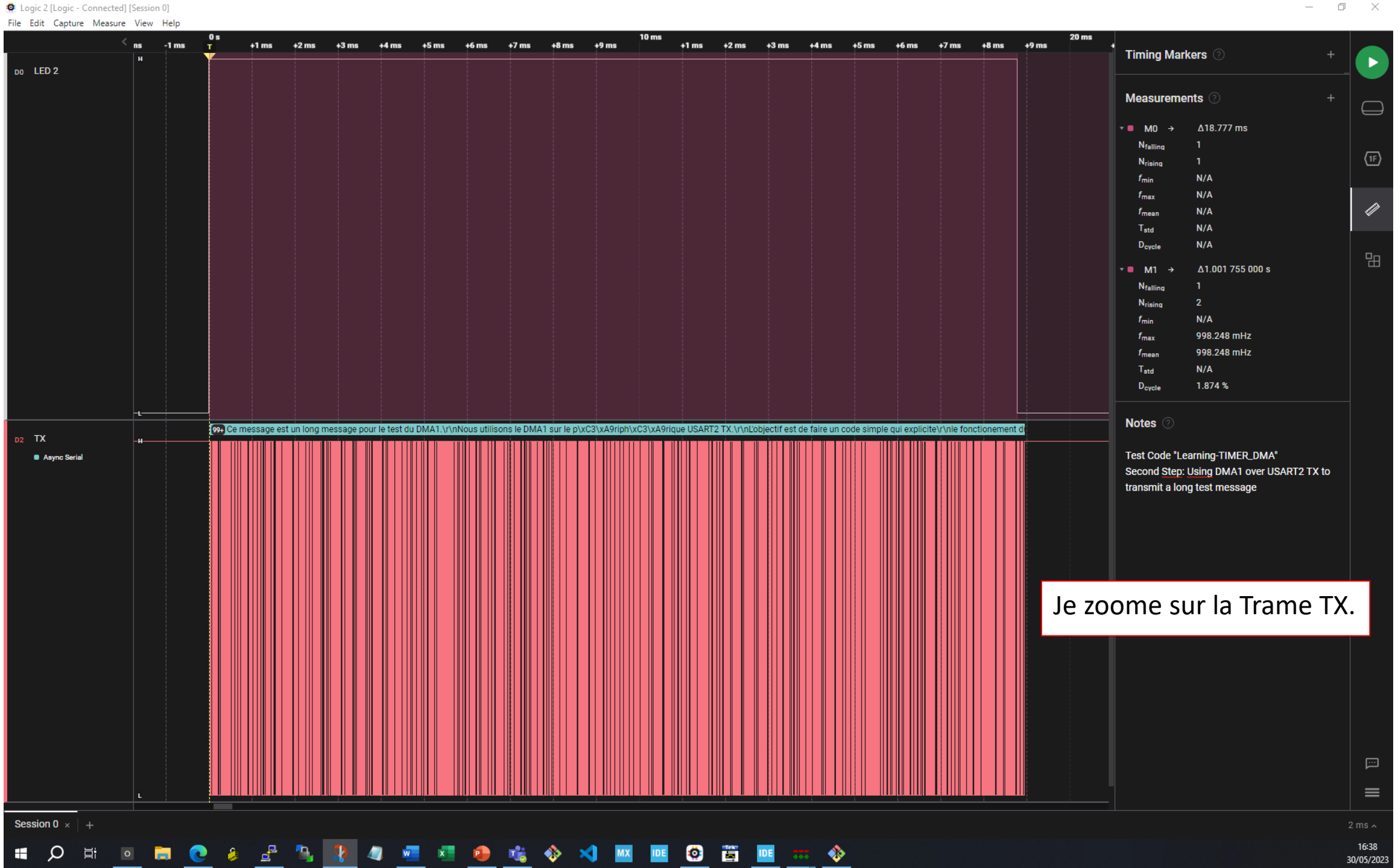
J’ai créé un tableau avec un long message et dans la boucle « While », j’envoie ce message toutes les secondes.

COM3 - PuTTY

```
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.

Ce message est un long message pour le test du DMA1.
Nous utilisons le DMA1 sur le périphérique USART2 TX.
L'objectif est de faire un code simple qui explicite
le fonctionement du DMA1 pour un novice comme moi.
```

Voici l’affichage du long message sur le terminal en liaison UART.

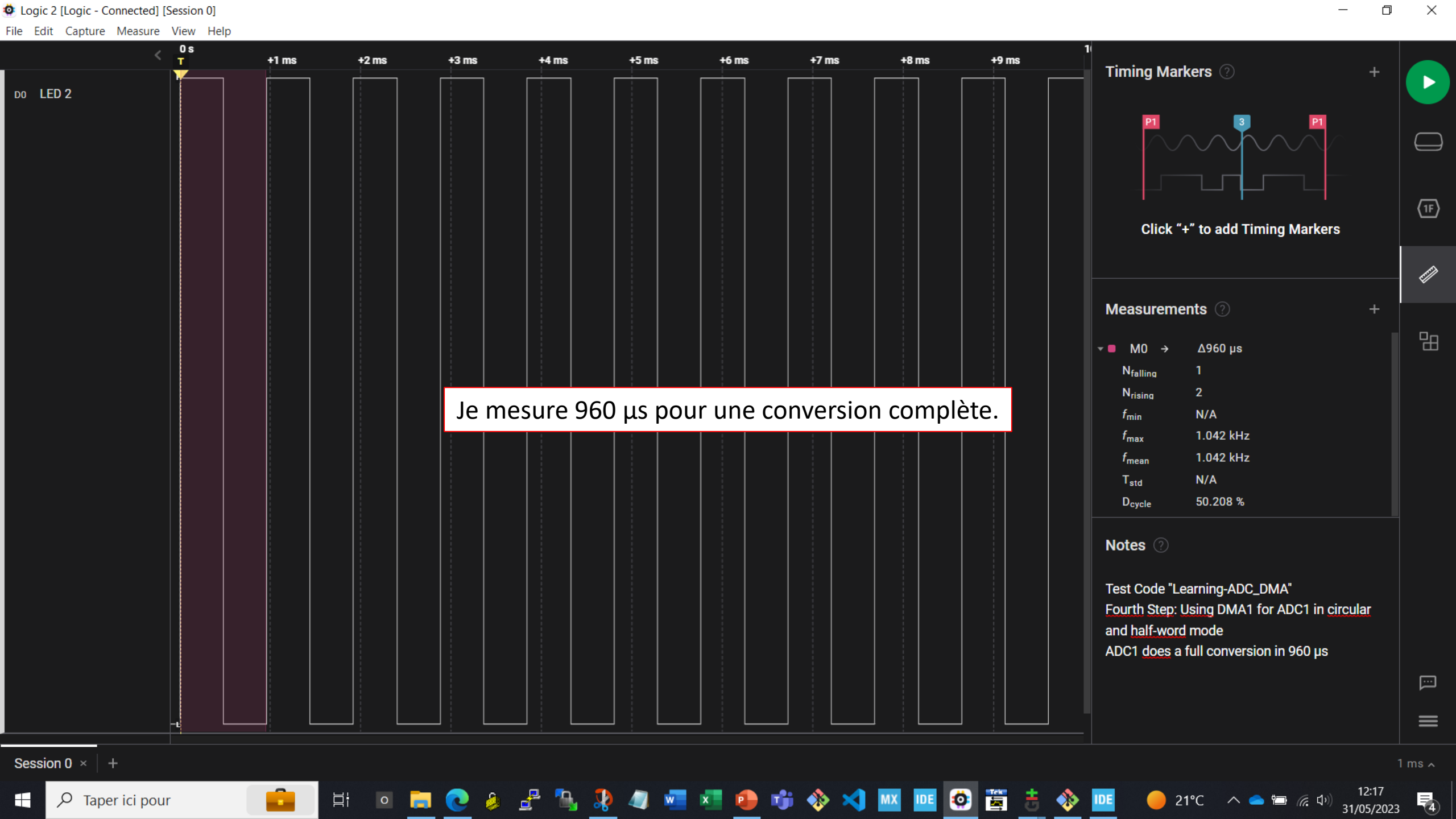

Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
D0 LED 2
D2 TX
Async Serial
Timing Markers
Measurements
M0 → Δ18.777 ms
Nfalling 1
Nrising 1
fmin N/A
fmax N/A
fmean N/A
Tstd N/A
Dcycle N/A
M1 → Δ1.001 755 000 s
Nfalling 1
Nrising 2
fmin N/A
fmax 998.248 mHz
fmean 998.248 mHz
Tstd N/A
Dcycle 1.874 %
Notes
Test Code "Learning-TIMER_DMA"
Second Step: Using DMA1 over USART2 TX to transmit a long test message
Je mesure bien 1 seconde entre chaque envoi.
La trame TX dure 18,8 ms et n'a pas d'impacte sur le temps d'envoi.
Session 0
89 ms
16:39
30/05/2023

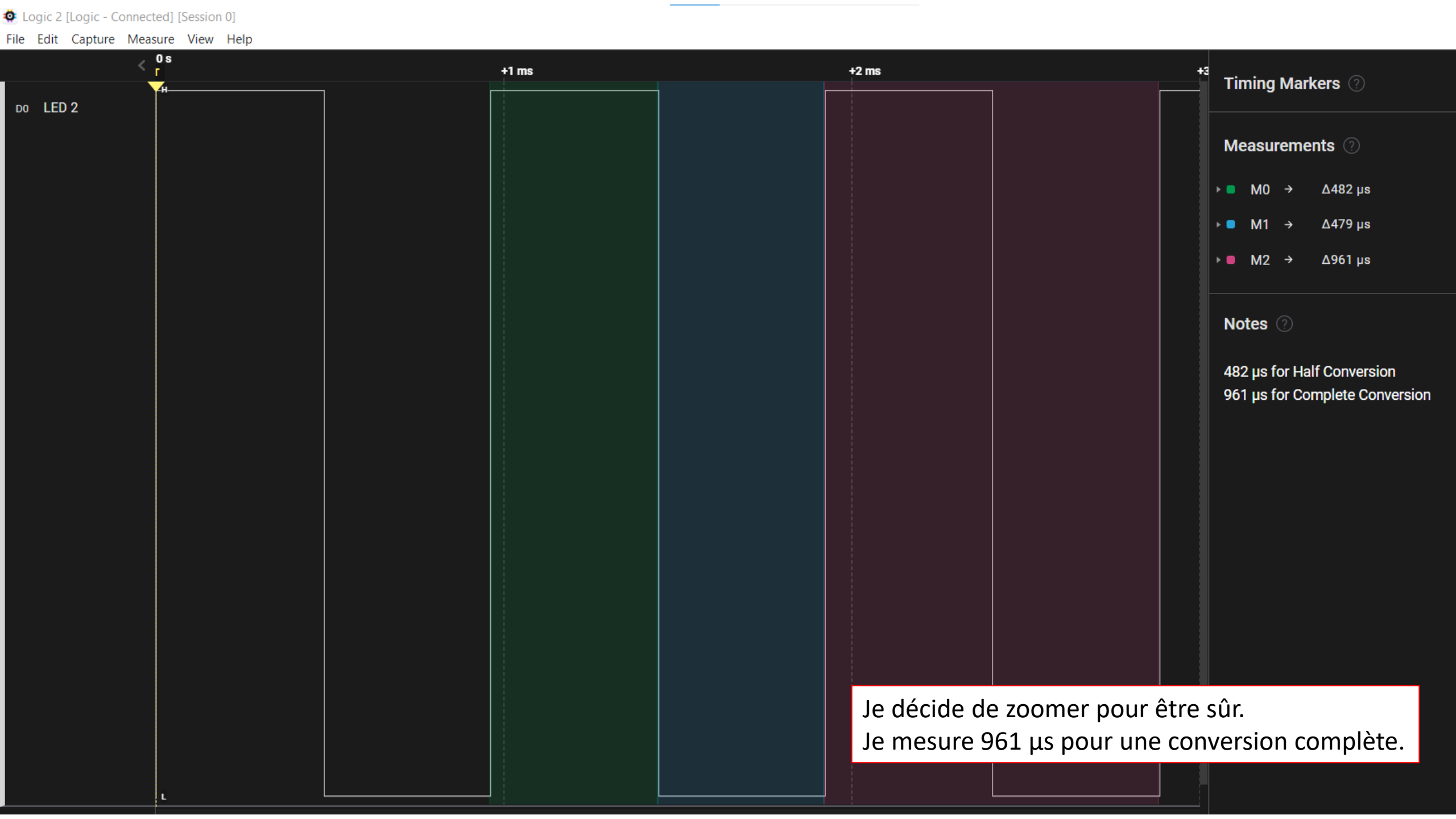

Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
D0 LED 2
D2 TX
Async Serial
99+ Ce message est un long message pour le test du DMA1.\r\nNous utilisons le DMA1 sur le p\xC3\xA9riph\xC3\xA9rique USART2 TX.\r\nL'objectif est de faire un code simple qui explicite\r\nle fonctionement d
Timing Markers
Measurements
M0 → Δ18.777 ms
N falling 1
N rising 1
f min N/A
f max N/A
f mean N/A
T std N/A
D cycle N/A
M1 → Δ1.001 755 000 s
N falling 1
N rising 2
f min N/A
f max 998.248 mHz
f mean 998.248 mHz
T std N/A
D cycle 1.874 %
Notes
Test Code "Learning-TIMER_DMA"
Second Step: Using DMA1 over USART2 TX to transmit a long test message
Je zoome sur la Trame TX.
Session 0
2 ms
16:38
30/05/2023

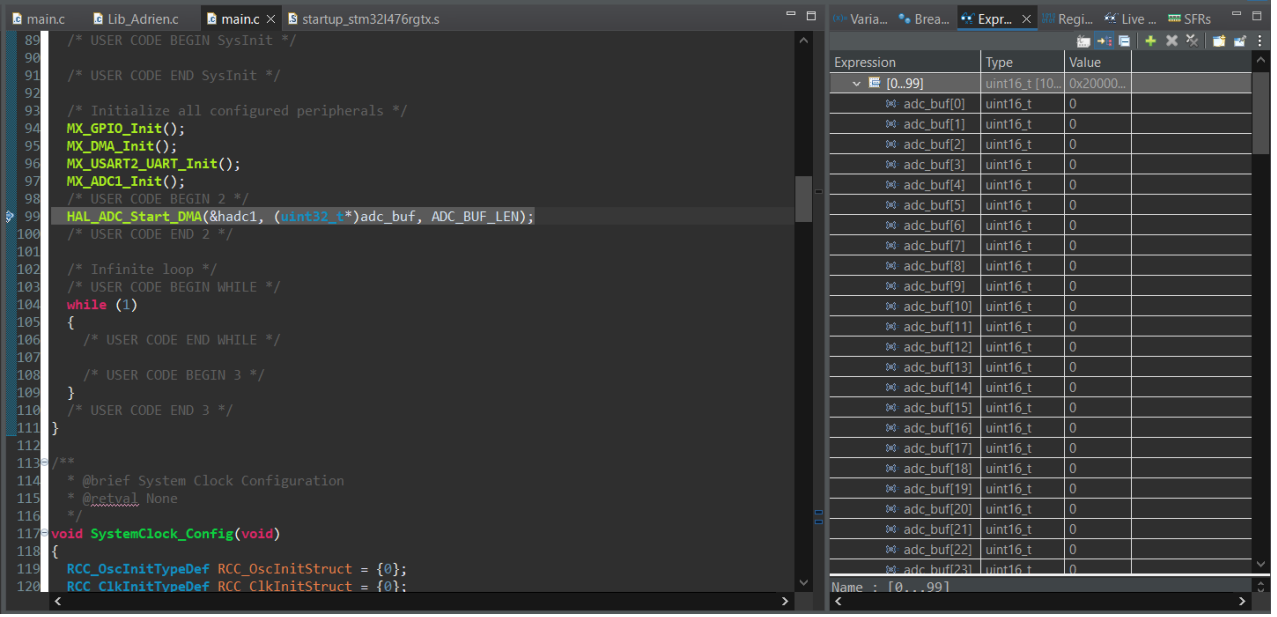
Troisième étape

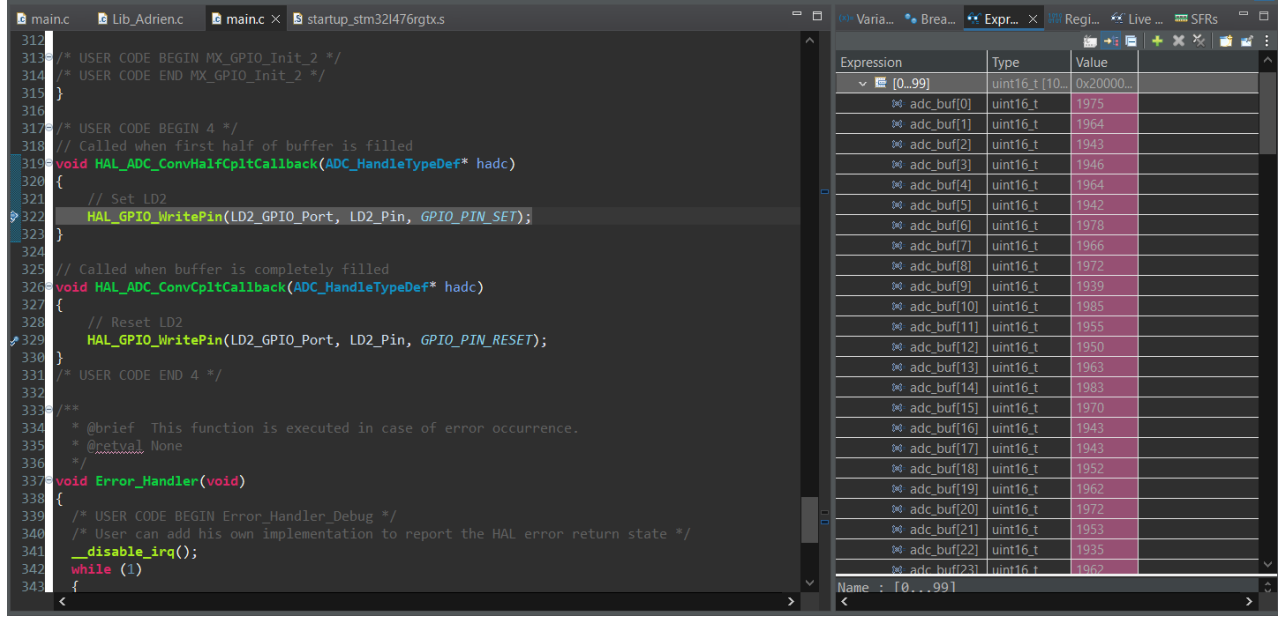
Cette fois-ci, j'utilise la DMA sur l'ADC en mode circulaire pour voir son impacte sur le débuggeur et mesurer le temps de conversion sur un buffer.

On crée un buffer de 4096 valeurs. L'ADC remplit le buffer de manière cyclique et en parallèle du CPU grâce à la DMA.

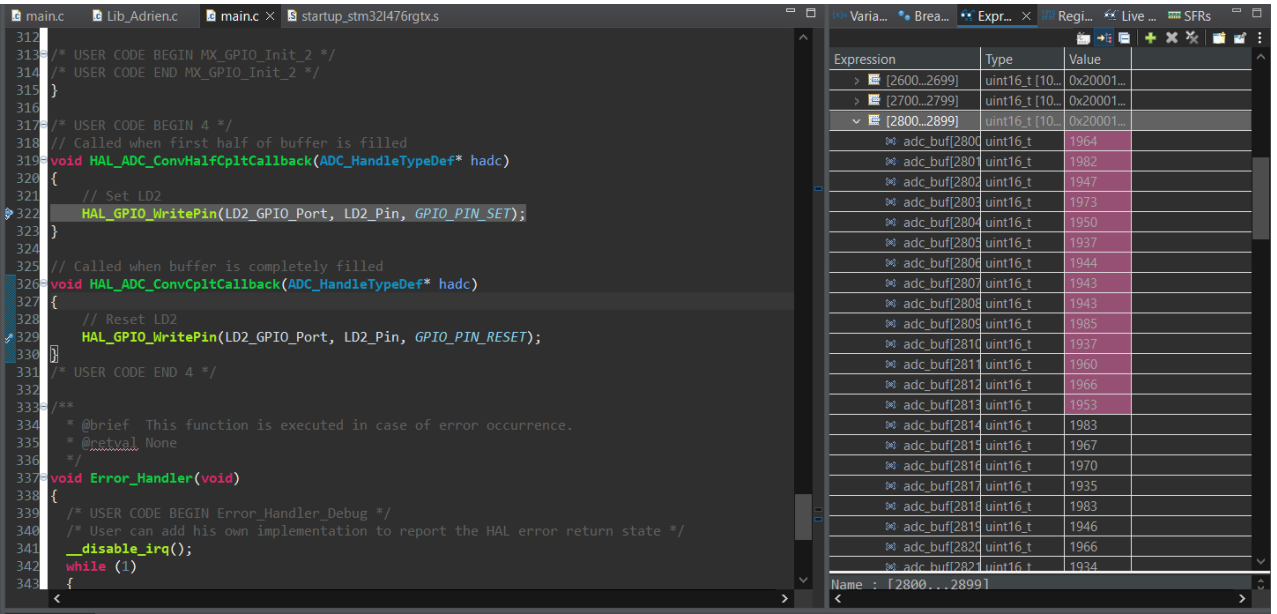

Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
0 s
T
+1 ms
+2 ms
+3 ms
+4 ms
+5 ms
+6 ms
+7 ms
+8 ms
+9 ms
D0 LED 2
Je mesure 960 µs pour une conversion complète.
Timing Markers
P1
3
P1
Click "+" to add Timing Markers
Measurements
M0 → Δ960 µs
N falling 1
N rising 2
f min N/A
f max 1.042 kHz
f mean 1.042 kHz
T std N/A
D cycle 50.208 %
Notes
Test Code "Learning-ADC_DMA"
Fourth Step: Using DMA1 for ADC1 in circular and half-word mode
ADC1 does a full conversion in 960 µs
Session 0
1 ms
Taper ici pour
21°C
12:17
31/05/2023

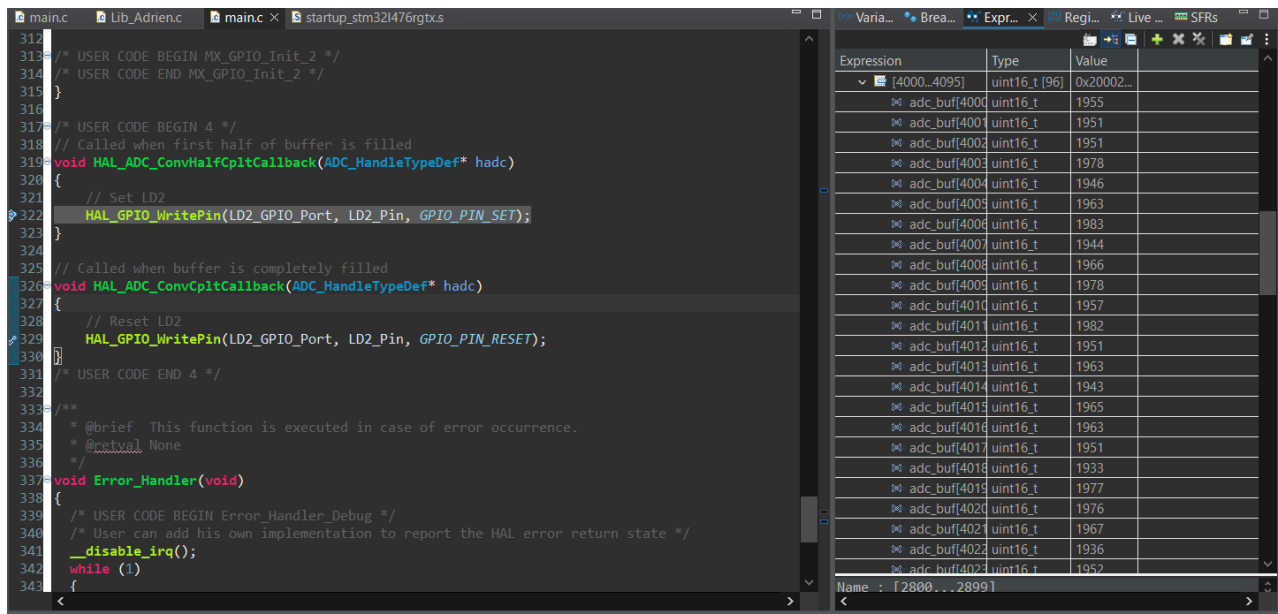

Logic 2 [Logic - Connected] [Session 0]
File Edit Capture Measure View Help
0 s
+1 ms
+2 ms
D0 LED 2
H
L
Timing Markers
Measurements
M0 → Δ482 µs
M1 → Δ479 µs
M2 → Δ961 µs
Notes
482 µs for Half Conversion
961 µs for Complete Conversion
Je décide de zoomer pour être sûr.
Je mesure 961 µs pour une conversion complète.

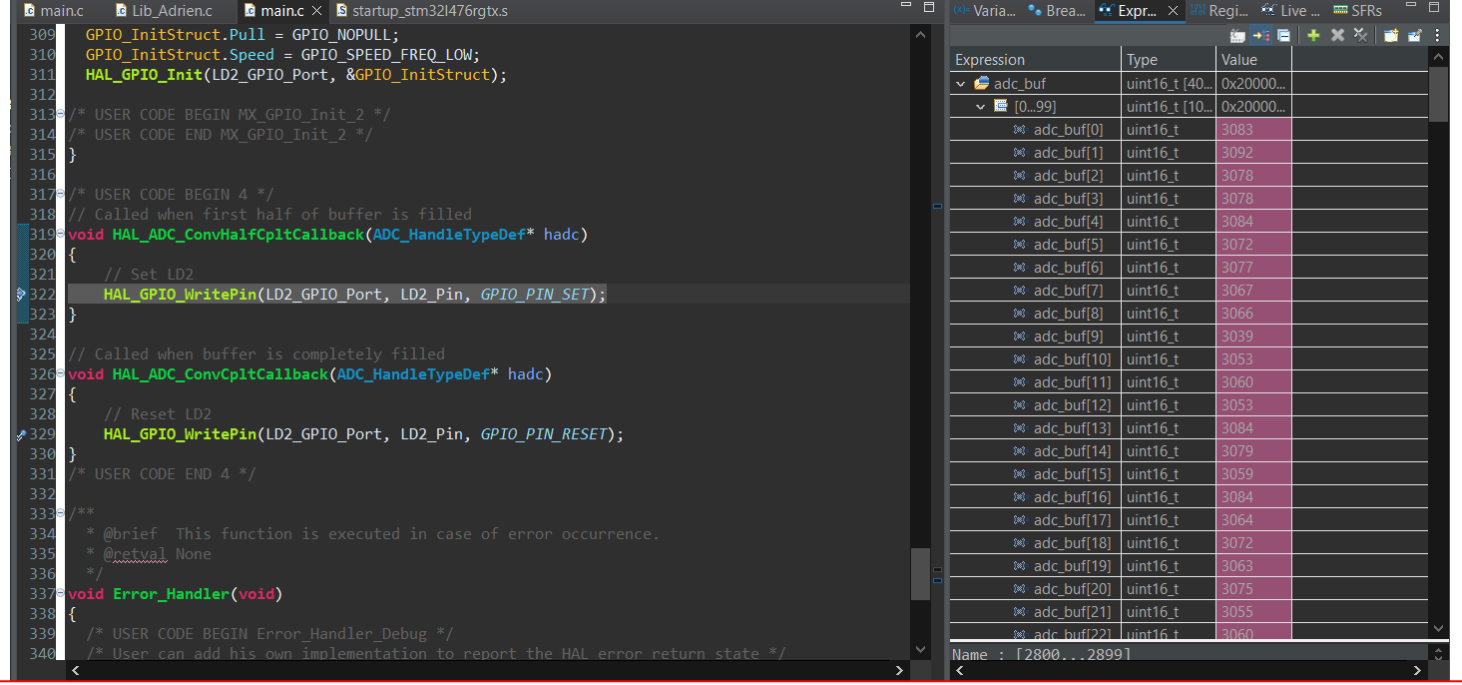




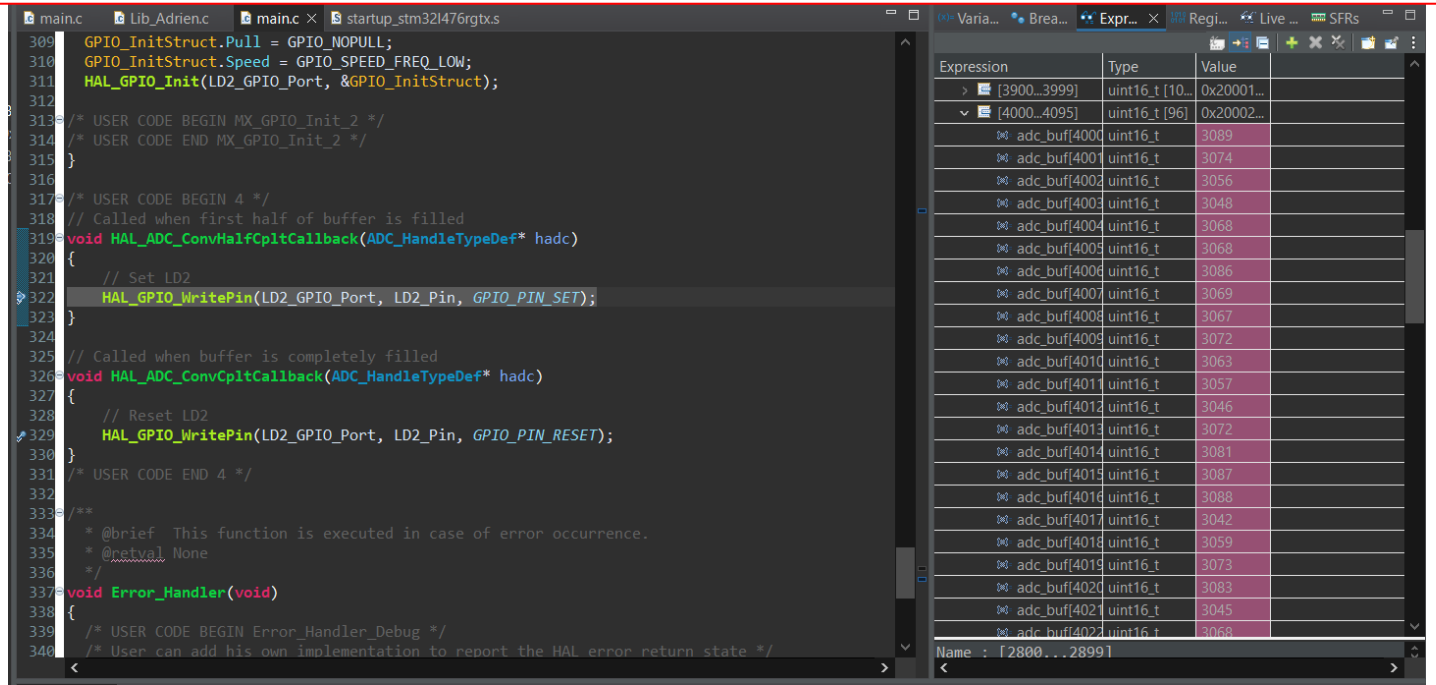
On passe en mode débugge pour voir le remplissage du buffer par la DMA.







On passe en mode débugge pour voir le remplissage du buffer par la DMA.



Conclusion

Grâce à ces exercices, j'ai appris à utiliser :

- Le logiciel Logic
- ADC en mode polling
- ADC en mode DMA circulaire
- L'UART en TX pour avoir un retour sur un terminal
- L'UART en TX en mode DMA
- L'avantage de l'utilisation de la DMA sur un buffer volumineux
- L'impacte de la DMA sur le mode Débug