EE2026: DIGITAL DESIGN

Academic Year 2020-2021, Semester 1

LAB 4: Sequential Circuits in Verilog - Part 2

OVERVIEW

This lab is a continuation of Lab 3. In this Lab 4, more applications of sequential circuits will be explored, and this will be the last lab practice before the EE2026 project.

The pre-requisites for this lab are:

- Knowing how to obtain slower clock frequencies from a faster one.
- Brief understanding of the purpose of a D-Flip-Flop (D-FF) in a sequential circuit.

This lab will cover the following:

- Schematic and design of a single pulse circuit, by using two D-FFs with a clock signal.
- Schematic and design of a debounced single pulse circuit, by using two D-FFs with a slow clock signal.
- Incrementing a counter value by 1 when a pushbutton is pressed.
- Multiplying a counter value by 2 when a pushbutton is pressed.

Tasks for this lab include:

- Simulating a single pulse circuit, whereby the D-FFs use a 100 MHz clock.
- Implementing a debounced single pulse circuit, by using a 3 Hz clock, on the Bays 3 development board.
- Observing patterns on the physical LED array, through counters that use the debounced single pulse signal.

GRADED ASSIGNMENT [LUMINUS SUBMISSION: TUESDAY 13th OCTOBER 2020, NOON]:

This is the final lab assignment to test whether you have properly mastered the technical contents from your three previous lab assignments. In this lab 4 assignment, the focus will be more on two very important components for the upcoming project: Integration and Logic

You are strongly encouraged to complete this assignment before the start of Week 8 (Monday 5th October 2020), as the focus will be on the EE2026 project from there on.

Further details are available at the end of this lab manual.

CREATING A SINGLE PULSE OUTPUT [Take 30 minutes before the lab to think how this can be done]

In this section, a single pulse output circuit will be created. The circuit from ***Figure 4.1***, consisting of two D-FFs and an AND gate, can be used to generate a single pulse output signal. The latter will be a synchronised logic true signal that lasts for the duration of one clock cycle.

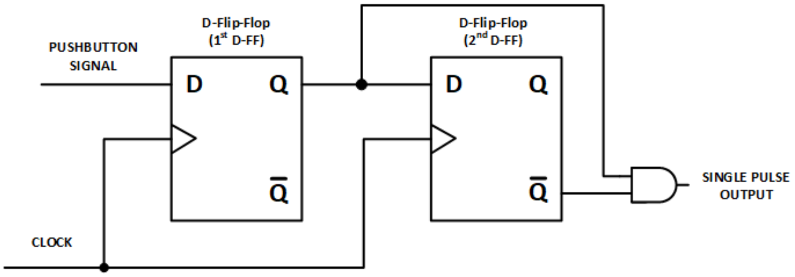


Figure 4.1: *Single pulse circuit*

The waveform for the single pulse circuit is as shown in ***Figure 4.2***

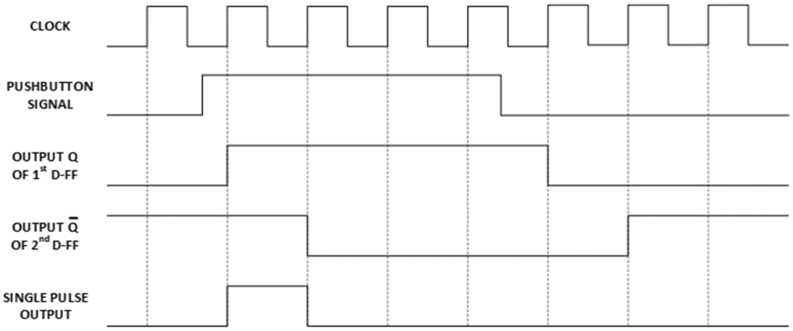


Figure 4.2: *Waveform from a single pulse circuit*

UNDERSTANDING | TASK 1

Through behavioural modelling in Verilog, the code for a positive-edge triggered D-flip-flop can be created:

Verilog code for a positive-edge triggered D-flip-flop

```
module my_dff(input DFF_CLOCK, D, output reg Q = 0);

    always @ (posedge DFF_CLOCK) begin
        Q <= D;
    end

endmodule
```

Using structural, dataflow and/or behavioural modelling, create the Verilog design for the circuit shown in ***Figure 4.1***. Create the simulation for the single pulse design module, with a simulated 100 MHz clock and appropriate stimuli, to confirm that a single pulse can be obtained. Take note of the time scale in the simulation window, and determine the duration of the single pulse output.

CREATING A DEBOUNCED SINGLE PULSE OUTPUT

On a physical device, the single pulse circuit can also be used to debounce mechanical switch signals. A slower clock, whose time period is longer than the bouncing of the switch signal, is required. Feeding this slower clock signal to the two D-FFs of a single pulse circuit will create a debounced single pulse output, as illustrated in ***Figure 4.3***

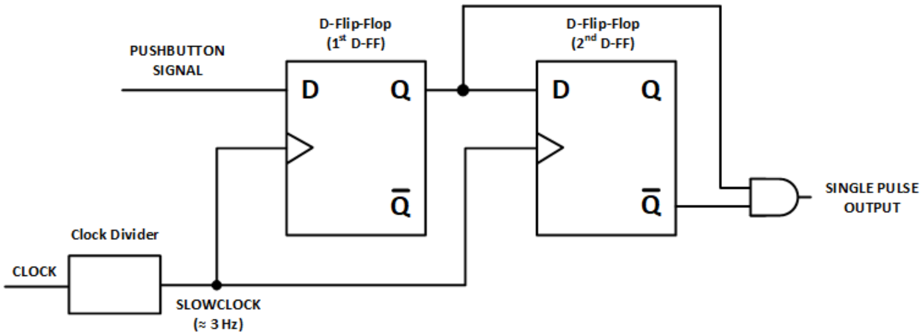


Figure 4.3: *Debounced single pulse circuit*

The waveform for the debounced single pulse circuit is as shown in ***Figure 4.4***

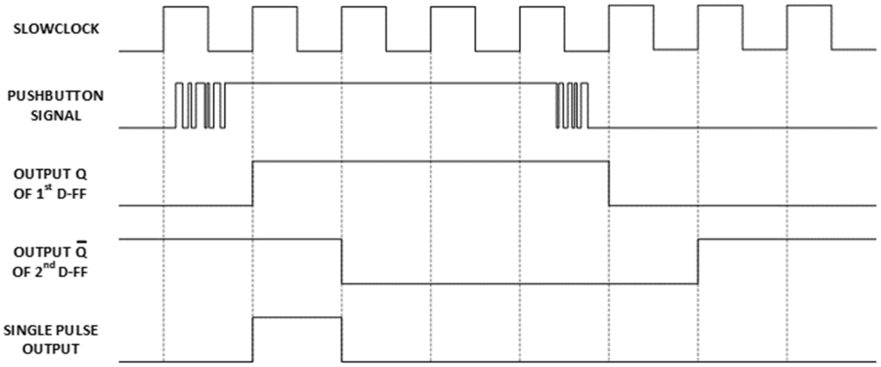


Figure 4.4: *Waveform from a debounced single pulse circuit*

UNDERSTANDING | TASK 2

Implement the circuit in ***Figure 4.3*** on the Basys 3 development board, by linking the debounced single pulse output to an LED. Press and hold the pushbutton to see if a single pulse of 1 clock cycle is obtained.

How much of simulated time would be required in order to see the single pulse output in the simulation waveform window?

Can you think of other ways of implementing a debounced single pulse output?

APPLYING THE DEBOUNCED SINGLE PULSE

From **UNDERSTANDING | TASK 2**, it is noted that no matter how long the pushbutton is pressed and held, a single synchronous pulse of only one clock cycle is created. This allows for the creation of a counter whose value can be made to increase by 1 each time a pushbutton is pressed, as such press would result in a positive edge of a single pulse.

UNDERSTANDING | TASK 3

Design an 8-bit counter that makes use of the debounced single pulse output, in order to increase its value by 1 each time the pushbutton is pressed, and for no matter how long the pushbutton is held. Implement the design on the Basys 3 development board, and observe how the physical LED array pattern changes whenever the pushbutton is pressed (and optionally held for a long time). Consider using the code below to detect the rising edge of the single pulse signal:

Partial Verilog code for a count that increments by one at the positive edge of the synchronised single pulse signal

```
// Initialise the initial 8-bit counter value to 8'b0000_0000
always @ (posedge SLOWCLOCK) begin
    if (SINGLE_PULSE_OUTPUT == 1) begin
        my_counter <= my_counter + 1;
    end
end
```

Important: *The clock used in the sensitivity list must be exactly the same clock that was used to create the single pulse output*

UNDERSTANDING | TASK 4

Shifting the values in a register to the left multiplies the register value by the base value. Create another design for a 16-bit counter, whereby pressing the pushbutton multiplies the value by 2 (Binary / Base-2 system). Implement the design on the Basys 3 development board. You may use the Verilog code template given below:

Partial Verilog code for a count that multiplies by 2 at the positive edge of the synchronised single pulse signal

```
// Initialise the initial 16-bit counter value to 16'h000C;
always @ (posedge SLOWCLOCK) begin
    if (SINGLE_PULSE_OUTPUT == 1) begin
        my_counter <= my_counter << 1;
    end
end
```

Record the values that you see from the LED array in the table below:

	Value from the Physical LED Array	Unsigned Decimal
Original Value	0000 0000 0000 1100	12
1st Pushbutton Press		
2nd Pushbutton Press		
3rd Pushbutton Press		
4th Pushbutton Press		
5th Pushbutton Press		
6th Pushbutton Press		
…		

What kind of pattern do you notice in the values?

____

What happens for pushbutton presses after the 13th and 14th ones?

____

UNDERSTANDING | TASK 5

Change your SLOWCLOCK to a 12 Hz clock. What do you notice when the pushbutton is pressed for the counter?

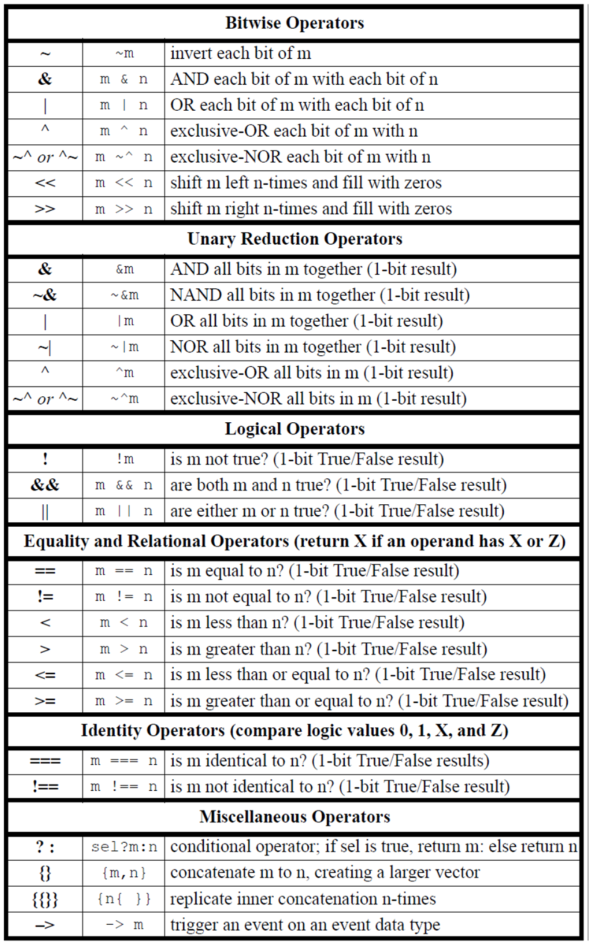
Change your SLOWCLOCK to a 25 MHz clock? What do you notice when the pushbutton is pressed for the counter?

GRADED POST-LAB ASSIGNMENT

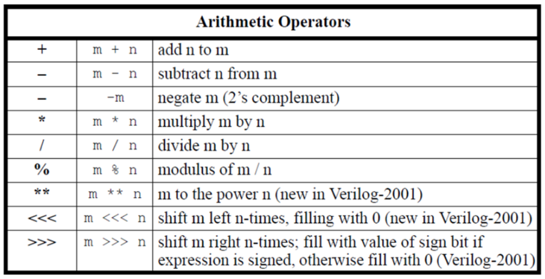
Read and understand all the sub-tasks before starting to work on your program.

Complete as much as possible, **in one working bitstream**, within the given deadline. It is much better to have a working program with some completed sub-tasks, instead of submitting a program without a working bitstream.

You may need a few of the operators listed below for the assignment (as well as for the upcoming EE2026 project):



Bitwise Operators		
~	~m	invert each bit of m
&	m & n	AND each bit of m with each bit of n
|	m | n	OR each bit of m with each bit of n
^	m ^ n	exclusive-OR each bit of m with n
~^ *or* ^~	m ~^ n	exclusive-NOR each bit of m with n
<<	m << n	shift m left n-times and fill with zeros
>>	m >> n	shift m right n-times and fill with zeros
Unary Reduction Operators		
&	&m	AND all bits in m together (1-bit result)
~&	~&m	NAND all bits in m together (1-bit result)
|	|m	OR all bits in m together (1-bit result)
~|	~|m	NOR all bits in m together (1-bit result)
^	^m	exclusive-OR all bits in m (1-bit result)
~^ *or* ^~	~^m	exclusive-NOR all bits in m (1-bit result)
Logical Operators		
!	!m	is m not true? (1-bit True/False result)
&&	m && n	are both m and n true? (1-bit True/False result)
||	m || n	are either m or n true? (1-bit True/False result)
Equality and Relational Operators (return X if an operand has X or Z)		
==	m == n	is m equal to n? (1-bit True/False result)
!=	m != n	is m not equal to n? (1-bit True/False result)
<	m < n	is m less than n? (1-bit True/False result)
>	m > n	is m greater than n? (1-bit True/False result)
<=	m <= n	is m less than or equal to n? (1-bit True/False result)
>=	m >= n	is m greater than or equal to n? (1-bit True/False result)
Identity Operators (compare logic values 0, 1, X, and Z)		
===	m === n	is m identical to n? (1-bit True/False results)
!==	m !== n	is m not identical to n? (1-bit True/False result)
Miscellaneous Operators		
? :	sel?m:n	conditional operator; if sel is true, return m: else return n
{}	{m,n}	concatenate m to n, creating a larger vector
{{}}	{n{ }}	replicate inner concatenation n-times
–>	-> m	trigger an event on an event data type

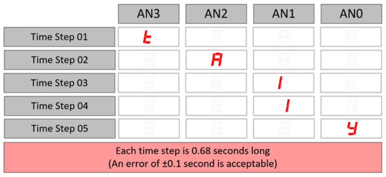


Arithmetic Operators		
+	m + n	add n to m
–	m - n	subtract n from m
–	-m	negate m (2's complement)
*	m * n	multiply m by n
/	m / n	divide m by n
%	m % n	modulus of m / n
**	m ** n	m to the power n (new in Verilog-2001)
<<<	m <<< n	shift m left n-times, filling with 0 (new in Verilog-2001)
>>>	m >>> n	shift m right n-times; fill with value of sign bit if expression is signed, otherwise fill with 0 (Verilog-2001)

It is required to complete and fully understand all the five UNDERSTANDING | TASK of lab 4 before starting this assignment. **The accuracy and responsiveness of the pushbuttons will be tested. High accuracy and high responsiveness are required.**

SUB-TASK A [TALLYING]

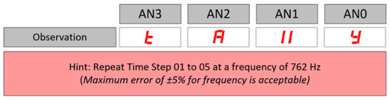
By default, the program starts with all the active-high switches set to ON (HIGH / 1), LEDs LD0 to LD15 all OFF, and all segments (7 segments + decimal point) of the seven-segment displays OFF. When this tallying sub-system starts, it is required to clearly observe the following sequence of 5 characters on specific seven-segment displays at the specified time steps:



	AN3	AN2	AN1	AN0
Time Step 01	t			
Time Step 02		A		
Time Step 03			I	
Time Step 04			I	
Time Step 05				Y

Each time step is 0.68 seconds long
(An error of ±0.1 second is acceptable)

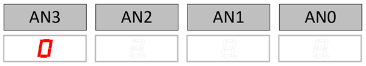
After Time Step 05 has shown the character “Y”, it is required to be able to clearly observe the following, until the user presses the up pushbutton (BTNU):



	AN3	AN2	AN1	AN0
Observation	t	A	II	Y

Hint: Repeat Time Step 01 to 05 at a frequency of 762 Hz
(Maximum error of ±5% for frequency is acceptable)

When the user presses the up pushbutton (BTNU) at least once, the following display is to be shown:

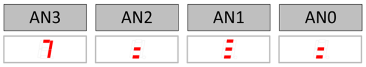


AN3	AN2	AN1	AN0
0			

Assume that AN0 to AN2 now represent separate rooms. Whenever a person enters the room, one additional horizontal bar is to be lit up, up to a maximum of 3 horizontal bars. The following actions indicate that a person is entering a room:

BTNL (Left pushbutton):	One additional person is entering room AN2
BTNC (Centre pushbutton):	One additional person is entering room AN1
BTNR (Right pushbutton):	One additional person is entering room AN0

At any time during the running of the tallying sub-system, AN3 indicates the total amount of persons in all three rooms, in hexadecimal number format. In this sub-system for tallying, assume that there will not be more than 3 persons per room. An example is shown below, with 2 persons inside room AN2, 3 persons inside room AN1, and 2 persons inside room AN0:



AN3	AN2	AN1	AN0
7	=	≡	=

The user may reset back to 0, the amount of person inside all the rooms by pressing the up pushbutton (BTNU) at any time during the running of the tallying sub-system.

SUB-TASK B [ACTIVE NOTIFICATIONS]

In order to prevent overcrowding and to distribute the amount of people inside the rooms evenly, the following two rules are to be implemented into the tallying sub-system:

(1) A person can only enter a room if the difference in number of persons inside the room will not be more than 1 (Difference can be 1) after entering the room, when compared to the other rooms

(2) No two persons can enter the same room one after the other

If any of the above two rules are broken, the person must not be allowed to enter the room.

An active notification in the form of a blinking decimal point and blinking horizontal bar(s) in that specific room the person was trying to enter should occur for a duration of 5.37 seconds (±0.5 second error is acceptable). The blinking decimal point and blinking horizontal bar(s) occur at a frequency of 6 Hz (Maximum error of ±5% in frequency is acceptable). If the blinking is happening, no person will be allowed to enter any of the rooms until the blinking stops. Example: If the segments in AN2 is blinking, none of three pushbuttons: BTNL, BTN and BTNR, should register any pushbutton presses.

SUB-TASK C [COMPLIANT EVACUATIONS]

In the event of a fire emergency, any persons in any rooms can exit at any time in any order. A fire emergency is triggered when SW0 is set to OFF (LOW / 0). The following actions indicate that a person is evacuating from a room:

BTNL (Left pushbutton):	One person is evacuating room AN2
BTNC (Centre pushbutton):	One person is evacuating room AN1
BTNR (Right pushbutton):	One person is evacuating room AN0

During such fire emergencies, when the total amount of persons in all three rooms combined is zero, the following message should be shown:

AN3	AN2	AN1	AN0
P	A	S	S

SUB-TASK D [DIGITAL INTEGRATED SYSTEM]

A complete system will now be required so that it can be used for the Digital Electronics Lab. Using what you have learnt in the SAFE system from lab 3, a complete Digital Integrated System for Tallying, Active Notifications and Compliant Evacuations is required. The complete SAFE-DISTANCE works as follows:

(1) As SW0 is ON by default when the program is uploaded, it is required to light up the 16 LEDs in increasing order, that is, from LD0 to LD15. Each additional LED need to wait 0.0836 seconds (±0.05 second error is acceptable) to light up. None of the LEDs must blink. They must all have fully ON LEDs.

(2) When all 16 LEDs (LD0 to LD15) are ON, sub-task A occurs

(3) Sub-task B, if it has been implemented, also runs as part of sub-task A

(4) Sub-task C is triggered when SW0 is set to OFF. In this complete SAFE-DISTANCE, it is furthermore required for the 16 LEDs to turn OFF in decreasing order (LD15 to LD0). Each additional LED need to wait 0.167 seconds (±0.05 second error is acceptable) to turn off. If the amount of oxygen runs out in the rooms, that is all the 16 LEDs are OFF, before all persons have left the room, the following message should be shown:

AN3	AN2	AN1	AN0
F	A	I	L

Assume that the program ends there when “PASS” or “FAIL” is shown. If there is a need to re-run the program, set SW0 to ON, and reload the bitstream to run the SAFE-DISTANCE.

LUMINUS SUBMISSION INSTRUCTIONS

- Complete as much required functionalities **as possible within the given deadline**, and ensure that your bitstream has been successfully generated and tested on your Basys 3 development board **BEFORE** archiving your Vivado workspace for LumiNUS upload. No working bitstream is equivalent to no marks (It is best to have some working functionalities / requirements, instead of not having any bitstream at all while trying all requirements)

- It is compulsory to archive your project in a compressed form without any simulation waveforms. In the uploaded archive, the codes (.v files) are important, not the waveforms (.wdb files). **The archive size should not exceed 4 MB in size for lab 4.** Follow the instructions given in the pdf: "Archive Project in Vivado 2018.02"

- **After** following the instructions in "**Archive Project in Vivado 2018.02**", rename your project archive as indicated in the appendix of this lab manual

- Upload to LumiNUS EE2026 -> Files -> Lab and Project - Materials and Submissions -> Lab 4 Submission

- Download your LumiNUS archive after uploading. **Unzip it / Extract all, and check if you can run your bitstream correctly**. No project files and no working bitstream is equivalent to losing all marks

- The LumiNUS upload must be completed by **Tuesday 13th October 2020, 12:00 P.M. (Noon)**. Do not plan to upload during the grace period of 2 hours

- Late submissions for lab 4 are not accepted. Complete as much required functionalities as possible within the given deadline and in one working program

Plagiarism is penalised with a 100% penalty for all SOURCES and RECIPIENTS

All past and future submissions, and marks, will be reviewed in greater detail, for any person found to have plagiarised

REMEMBER TO NAME YOUR SUBMISSION IN THE REQUIRED FORMAT

ALL THE SUBMISSION INSTRUCTIONS LISTED ABOVE WILL AFFECT YOUR GRADES!

APPENDIX (Renaming submissions just before LumiNUS upload):

It is **compulsory to rename your project archive**, just before LumiNUS upload, as indicated in the table below. Copy your respective “Archive Naming” for the archive, and then replace the “xxxxxxxxx” with your student ID number. Do not change any other part of the naming, except the “xxxxxxxxx”.

Submission example for project archive: L4_Fri_AM_Alan Turing_Archive_A0131086Z

Name	Archive Naming
A AKIL AHAMED	L4_Fri_AM_A AKIL AHAMED_Archive_XXXXXXXXX
Abdul Hadi Bin Abdul Samad	L4_Fri_AM_Abdul Hadi Bin Abdul_Archive_XXXXXXXXX
Adabelle Lim Ru Leng	L4_Fri_AM_Adabelle Lim Ru Leng_Archive_XXXXXXXXX
Alfred Wrong Jia Qing	L4_Fri_AM_Alfred Wrong Jia Qin_Archive_XXXXXXXXX
Alvin Goh Jia Hao	L4_Fri_AM_Alvin Goh Jia Hao_Archive_XXXXXXXXX
Alvinci Merquita	L4_Wed_AM_Alvinci Merquita_Archive_XXXXXXXXX
ANG KENG SIANG	L4_Wed_AM_ANG KENG SIANG_Archive_XXXXXXXXX
Aryl Ng Shen Le	L4_Wed_AM_Aryl Ng Shen Le_Archive_XXXXXXXXX
Au Yuan Xian	L4_Fri_AM_Au Yuan Xian_Archive_XXXXXXXXX
Bai Xiaoru	L4_Fri_AM_Bai Xiaoru_Archive_XXXXXXXXX
Bryan Yu Cheng You	L4_Fri_AM_Bryan Yu Cheng You_Archive_XXXXXXXXX
Chai Wei Lynthia	L4_Fri_AM_Chai Wei Lynthia_Archive_XXXXXXXXX
Cheang Zhi Yi Jordan	L4_Fri_AM_Cheang Zhi Yi Jordan_Archive_XXXXXXXXX
Chee Poh Hock	L4_Wed_AM_Chee Poh Hock_Archive_XXXXXXXXX
Cheng Wei Qiao	L4_Fri_AM_Cheng Wei Qiao_Archive_XXXXXXXXX
Cheung Po Rui Bryan	L4_Fri_AM_Cheung Po Rui Bryan_Archive_XXXXXXXXX
CHONG LEE TENG VALENCIA	L4_Wed_AM_CHONG LEE TENG VALEN_Archive_XXXXXXXXX
Davian Chan Sze Peng	L4_Fri_AM_Davian Chan Sze Peng_Archive_XXXXXXXXX
David Michael Woodside	L4_Fri_AM_David Michael Woodsi_Archive_XXXXXXXXX
ELJER CHUA	L4_Fri_AM_ELJER CHUA_Archive_XXXXXXXXX
FANG XINJIA	L4_Fri_AM_FANG XINJIA_Archive_XXXXXXXXX
Fidel Tan Yan Sheng	L4_Fri_AM_Fidel Tan Yan Sheng_Archive_XXXXXXXXX
Foo Fang Kiang	L4_Fri_AM_Foo Fang Kiang_Archive_XXXXXXXXX
Gao Zhixuan	L4_Fri_AM_Gao Zhixuan_Archive_XXXXXXXXX
Giam Xiong Yao	L4_Fri_AM_Giam Xiong Yao_Archive_XXXXXXXXX
Gillian Ho Xin Ying	L4_Fri_AM_Gillian Ho Xin Ying_Archive_XXXXXXXXX
Goh Jia Hong Edwin	L4_Wed_AM_Goh Jia Hong Edwin_Archive_XXXXXXXXX
Guinne Teresa Sng Yu Lin	L4_Fri_AM_Guinne Teresa Sng Yu_Archive_XXXXXXXXX
Hariharan Hadrian S/O Subramaniam	L4_Fri_AM_Hariharan Hadrian S_Archive_XXXXXXXXX
HO MING JUN	L4_Wed_AM_HO MING JUN_Archive_XXXXXXXXX
Ho Yi Shu Keon	L4_Wed_AM_Ho Yi Shu Keon_Archive_XXXXXXXXX
Hou Yinjiayi	L4_Fri_AM_Hou Yinjiayi_Archive_XXXXXXXXX
Ian Isaiah Tan Jun Wei	L4_Fri_AM_Ian Isaiah Tan Jun W_Archive_XXXXXXXXX
Jacob Zhang Zhiqiang	L4_Wed_AM_Jacob Zhang Zhiqiang_Archive_XXXXXXXXX
JEROME TEO SZE YONG	L4_Wed_AM_JEROME TEO SZE YONG_Archive_XXXXXXXXX
Jonathan Ang Xu Wen	L4_Wed_AM_Jonathan Ang Xu Wen_Archive_XXXXXXXXX
JONATHAN KHOO TENG YANG	L4_Fri_AM_JONATHAN KHOO TENG Y_Archive_XXXXXXXXX
Kabeta Takuma	L4_Wed_AM_Kabeta Takuma_Archive_XXXXXXXXX
Khoo Wu Jian Samuel	L4_Wed_AM_Khoo Wu Jian Samuel_Archive_XXXXXXXXX
KIM JOOHWAN	L4_Fri_AM_KIM JOOHWAN_Archive_XXXXXXXXX
Lau Wai Kit	L4_Wed_AM_Lau Wai Kit_Archive_XXXXXXXXX
LEE KE HUI	L4_Wed_AM_LEE KE HUI_Archive_XXXXXXXXX
Lee Shao Yu	L4_Wed_AM_Lee Shao Yu_Archive_XXXXXXXXX
Lek Ju Ying	L4_Wed_AM_Lek Ju Ying_Archive_XXXXXXXXX
Leong Ka Weng, Rachelle	L4_Fri_AM_Leong Ka Weng Rache_Archive_XXXXXXXXX
LEW POH CHEN, DOUGLAS	L4_Fri_AM_LEW POH CHEN DOUGLA_Archive_XXXXXXXXX
Long Deng Jie	L4_Wed_AM_Long Deng Jie_Archive_XXXXXXXXX
Markus Lim Yi Qin	L4_Wed_AM_Markus Lim Yi Qin_Archive_XXXXXXXXX
Mohamad Adam Bin Mohamad Yazid	L4_Wed_AM_Mohamad Adam Bin Moh_Archive_XXXXXXXXX
Muhammad Irfan Bin Zakaria	L4_Wed_AM_Muhammad Irfan Bin Z_Archive_XXXXXXXXX
Myat Thwe Naing	L4_Wed_AM_Myat Thwe Naing_Archive_XXXXXXXXX
Ng Etek	L4_Wed_AM_Ng Etek_Archive_XXXXXXXXX
Noorhakim Bin Jasman	L4_Wed_AM_Noorhakim Bin Jasman_Archive_XXXXXXXXX
NUR SYADIYAH BTE LUTFI	L4_Fri_AM_NUR SYADIYAH BTE LUT_Archive_XXXXXXXXX
ONG WEI SHENG	L4_Fri_AM_ONG WEI SHENG_Archive_XXXXXXXXX
PANG JUN WEN, ADRIC	L4_Fri_AM_PANG JUN WEN ADRIC_Archive_XXXXXXXXX
PUN ZE YONG	L4_Wed_AM_PUN ZE YONG_Archive_XXXXXXXXX
Qiang Zhuang	L4_Wed_AM_Qiang Zhuang_Archive_XXXXXXXXX
QIU YI WEN	L4_Fri_AM_QIU YI WEN_Archive_XXXXXXXXX
R M RAAJAMANI	L4_Fri_AM_R M RAAJAMANI_Archive_XXXXXXXXX
RIZAVUR RAHMAN FASLUR RAHMAN	L4_Fri_AM_RIZAVUR RAHMAN FASLU_Archive_XXXXXXXXX
Ryan Tan Jun Hao	L4_Wed_AM_Ryan Tan Jun Hao_Archive_XXXXXXXXX
Saw Wee Kiat	L4_Wed_AM_Saw Wee Kiat_Archive_XXXXXXXXX
SIM BOWEN	L4_Fri_AM_SIM BOWEN_Archive_XXXXXXXXX
TAM LI NA	L4_Fri_AM_TAM LI NA_Archive_XXXXXXXXX
Tan Javen	L4_Wed_AM_Tan Javen_Archive_XXXXXXXXX
Tan Kai Hao Andrew	L4_Wed_AM_Tan Kai Hao Andrew_Archive_XXXXXXXXX
Tan Suet Ying	L4_Fri_AM_Tan Suet Ying_Archive_XXXXXXXXX
Tan Yeung Ming Sean Eugene	L4_Wed_AM_Tan Yeung Ming Sean _Archive_XXXXXXXXX
Teoh Yi Zheng	L4_Fri_AM_Teoh Yi Zheng_Archive_XXXXXXXXX
Tey Zi Le	L4_Wed_AM_Tey Zi Le_Archive_XXXXXXXXX
Thet Ke Min, Sonia	L4_Wed_AM_Thet Ke Min Sonia_Archive_XXXXXXXXX
Trina Wern Qin Rong	L4_Fri_AM_Trina Wern Qin Rong_Archive_XXXXXXXXX
Venessa Chee Li Lin	L4_Fri_AM_Venessa Chee Li Lin_Archive_XXXXXXXXX
WANG HUA CHEN	L4_Fri_AM_WANG HUA CHEN_Archive_XXXXXXXXX
Wee Cheng Yuan Andrew	L4_Wed_AM_Wee Cheng Yuan Andre_Archive_XXXXXXXXX
Wee Xin Ze	L4_Wed_AM_Wee Xin Ze_Archive_XXXXXXXXX
Yeo Zhong Kang Dennis	L4_Wed_AM_Yeo Zhong Kang Denni_Archive_XXXXXXXXX
ZHONG SHUHAO	L4_Fri_AM_ZHONG SHUHAO_Archive_XXXXXXXXX