



**Πανεπιστήμιο Πατρών**



***Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας  
Υπολογιστών***

**Εργαστήριο Σχεδίασης Ολοκληρωμένων Κυκλωμάτων**

**Σχεδιασμός Ολοκληρωμένων Συστημάτων**

**Χειμερινό Εξάμηνο 2023**

**Εργαστήριο 2**

## Άσκηση 1: Βαρελοειδής ολισθητής

Ο βαρελοειδής ολισθητής (Barrel Shifter) είναι ένα σχετικά μεγάλο κύκλωμα με σημαντική καθυστέρηση. Στη γενική του μορφή μπορεί να ολισθαίνει το σήμα εισόδου εκτελώντας rotate/arithmetical/logical shift δεξιά ή αριστερά για μεταβλητό αριθμών θέσεων. Για το λόγο αυτό έχει επιπλέον εισόδους που καθορίζουν το είδος της ολισθησης (rotate/arithmetical/logical), την κατεύθυνση (δεξιά/αριστερά) και το πλήθος των θέσεων ολισθησης.

Περιγράψτε σε VHDL ένα βαρελοειδή ολισθητή που εκτελεί μόνο δεξιά ολισθηση και έχει:

A) μία είσοδο  $a$  (8-bit),

B) μία είσοδο  $lar$  (2-bit) όπου η τιμή “00” σημαίνει λογική ολισθηση (logical shift), η “01” αριθμητική ολισθηση (arithmetic shift) και οι τιμές “10”/“11” περιστροφή (rotation).

Γ) μία είσοδο  $amt$  (3-bit) που δηλώνει των αριθμών των θέσεων ολισθησης/περιστροφής. Δηλαδή, “000” καμία θέση, “001” μία θέση “010” δύο θέσεις κ.ο.κ.

Δ) μία έξοδο  $y$  (8-bit).

Όλα τα σήματα εισόδου και εξόδου είναι τύπου `std_logic`. Αν μπορείτε, δώστε δύο διαφορετικές περιγραφές για το παραπάνω κύκλωμα.

## Άσκηση 2: Μετατροπές Χαρακτήρων από HEX σε ASCII

Να σχεδιάσετε και να υλοποιήσετε σύστημα μετατροπής από HEX σε ASCII. Το σύστημα θα δέχεται ως είσοδο ένα χαρακτήρα HEX (4-bit) και θα βγάζει στην έξοδο τον αντίστοιχο ASCII κωδικό του (8-bit). Συγκεκριμένα, θεωρίστε ότι η είσοδος αντιστοιχεί στους αριθμούς 0-9 και στους κεφαλαίους χαρακτήρες A-F.

**A.** Δώστε μία λύση με τη χρήση μίας μνήμης (Look-Up-Table, LUT).

**B.** Παρατηρώντας την ιδιαιτερότητα των ASCII κωδικών, δώστε μία βελτιωμένη λύση μικρότερης μνήμης (LUT).

**C.** Με βάση τις προηγούμενες παρατηρήσεις δώστε μία λύση χωρίς τη χρήση LUT.

Μην συγχέετε το LUT που αναφέρεται παραπάνω με τα εσωτερικά LUTs του FPGA. Ως LUT, νοείται η «φυσική» σημασία των CASE/WHEN/WITH statements. Χρησιμοποιήστε απλές δομές συνιτρέχοντος κώδικα με WHEN.

Συμβουλευτείτε τον παρακάτω πίνακα αντιστοίχισης κωδικών ASCII.)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	<b>NUL</b> (null)	32	20	040	&#32;	<b>Space</b>	64	40	100	&#64;	<b>B</b>	96	60	140	&#96;	<b>`</b>
1	1	001	<b>SOH</b> (start of heading)	33	21	041	&#33;	<b>!</b>	65	41	101	&#65;	<b>A</b>	97	61	141	&#97;	<b>a</b>
2	2	002	<b>STX</b> (start of text)	34	22	042	&#34;	<b>"</b>	66	42	102	&#66;	<b>B</b>	98	62	142	&#98;	<b>b</b>
3	3	003	<b>ETX</b> (end of text)	35	23	043	&#35;	<b>#</b>	67	43	103	&#67;	<b>C</b>	99	63	143	&#99;	<b>c</b>
4	4	004	<b>EOT</b> (end of transmission)	36	24	044	&#36;	<b>\$</b>	68	44	104	&#68;	<b>D</b>	100	64	144	&#100;	<b>d</b>
5	5	005	<b>ENQ</b> (enquiry)	37	25	045	&#37;	<b>%</b>	69	45	105	&#69;	<b>E</b>	101	65	145	&#101;	<b>e</b>
6	6	006	<b>ACK</b> (acknowledge)	38	26	046	&#38;	<b>&amp;</b>	70	46	106	&#70;	<b>F</b>	102	66	146	&#102;	<b>f</b>
7	7	007	<b>BEL</b> (bell)	39	27	047	&#39;	<b>'</b>	71	47	107	&#71;	<b>G</b>	103	67	147	&#103;	<b>g</b>
8	8	010	<b>BS</b> (backspace)	40	28	050	&#40;	<b>(</b>	72	48	110	&#72;	<b>H</b>	104	68	150	&#104;	<b>h</b>
9	9	011	<b>TAB</b> (horizontal tab)	41	29	051	&#41;	<b>)</b>	73	49	111	&#73;	<b>I</b>	105	69	151	&#105;	<b>i</b>
10	A	012	<b>LF</b> (NL line feed, new line)	42	2A	052	&#42;	<b>*</b>	74	4A	112	&#74;	<b>J</b>	106	6A	152	&#106;	<b>j</b>
11	B	013	<b>VT</b> (vertical tab)	43	2B	053	&#43;	<b>+</b>	75	4B	113	&#75;	<b>K</b>	107	6B	153	&#107;	<b>k</b>
12	C	014	<b>FF</b> (NP form feed, new page)	44	2C	054	&#44;	<b>,</b>	76	4C	114	&#76;	<b>L</b>	108	6C	154	&#108;	<b>l</b>
13	D	015	<b>CR</b> (carriage return)	45	2D	055	&#45;	<b>-</b>	77	4D	115	&#77;	<b>M</b>	109	6D	155	&#109;	<b>m</b>
14	E	016	<b>SO</b> (shift out)	46	2E	056	&#46;	<b>.</b>	78	4E	116	&#78;	<b>N</b>	110	6E	156	&#110;	<b>n</b>
15	F	017	<b>SI</b> (shift in)	47	2F	057	&#47;	<b>/</b>	79	4F	117	&#79;	<b>O</b>	111	6F	157	&#111;	<b>o</b>
16	10	020	<b>DLE</b> (data link escape)	48	30	060	&#48;	<b>0</b>	80	50	120	&#80;	<b>P</b>	112	70	160	&#112;	<b>p</b>
17	11	021	<b>DC1</b> (device control 1)	49	31	061	&#49;	<b>1</b>	81	51	121	&#81;	<b>Q</b>	113	71	161	&#113;	<b>q</b>
18	12	022	<b>DC2</b> (device control 2)	50	32	062	&#50;	<b>2</b>	82	52	122	&#82;	<b>R</b>	114	72	162	&#114;	<b>r</b>
19	13	023	<b>DC3</b> (device control 3)	51	33	063	&#51;	<b>3</b>	83	53	123	&#83;	<b>S</b>	115	73	163	&#115;	<b>s</b>
20	14	024	<b>DC4</b> (device control 4)	52	34	064	&#52;	<b>4</b>	84	54	124	&#84;	<b>T</b>	116	74	164	&#116;	<b>t</b>
21	15	025	<b>NAK</b> (negative acknowledge)	53	35	065	&#53;	<b>5</b>	85	55	125	&#85;	<b>U</b>	117	75	165	&#117;	<b>u</b>
22	16	026	<b>SYN</b> (synchronous idle)	54	36	066	&#54;	<b>6</b>	86	56	126	&#86;	<b>V</b>	118	76	166	&#118;	<b>v</b>
23	17	027	<b>ETB</b> (end of trans. block)	55	37	067	&#55;	<b>7</b>	87	57	127	&#87;	<b>W</b>	119	77	167	&#119;	<b>w</b>
24	18	030	<b>CAN</b> (cancel)	56	38	070	&#56;	<b>8</b>	88	58	130	&#88;	<b>X</b>	120	78	170	&#120;	<b>x</b>
25	19	031	<b>EM</b> (end of medium)	57	39	071	&#57;	<b>9</b>	89	59	131	&#89;	<b>Y</b>	121	79	171	&#121;	<b>y</b>
26	1A	032	<b>SUB</b> (substitute)	58	3A	072	&#58;	<b>:</b>	90	5A	132	&#90;	<b>Z</b>	122	7A	172	&#122;	<b>z</b>
27	1B	033	<b>ESC</b> (escape)	59	3B	073	&#59;	<b>:</b>	91	5B	133	&#91;	<b>[</b>	123	7B	173	&#123;	<b>{</b>
28	1C	034	<b>FS</b> (file separator)	60	3C	074	&#60;	<b>&lt;</b>	92	5C	134	&#92;	<b>\</b>	124	7C	174	&#124;	<b> </b>
29	1D	035	<b>GS</b> (group separator)	61	3D	075	&#61;	<b>=</b>	93	5D	135	&#93;	<b>]</b>	125	7D	175	&#125;	<b>}</b>
30	1E	036	<b>RS</b> (record separator)	62	3E	076	&#62;	<b>&gt;</b>	94	5E	136	&#94;	<b>^</b>	126	7E	176	&#126;	<b>~</b>
31	1F	037	<b>US</b> (unit separator)	63	3F	077	&#63;	<b>?</b>	95	5F	137	&#95;	<b>_</b>	127	7F	177	&#127;	<b>DEL</b>

Source: [www.LookupTables.com](http://www.LookupTables.com)

### Άσκηση 3: Υπολογισμός ψηφίου ισοτιμίας και διάταξη “1s”

Θεωρείστε κύκλωμα με μια είσοδο των N ψηφίων τύπου `std_logic` και δύο εξόδους – μία του ενός ψηφίου και μία των N ψηφίων– επίσης τύπου `std_logic`. Το κύκλωμα εκτελεί τις ακόλουθες λειτουργίες:

- Υπολογίζει το ψηφίο ισοτιμίας (για περιττή ισοτιμία) της εισόδου το οποίο και επιστρέφει στην έξοδο
- Αναδιατάσσει τα ψηφία εισόδου τοποθετώντας τους “1” στις δεξιότερες θέσεις και τα μηδενικά στα αριστερά και επιστρέφει τη νέα λέξη στην έξοδο.

Γράψτε κώδικα VHDL για το παραπάνω κύκλωμα, ο οποίος πρέπει να είναι παραμετρικός ως προς το πλήθος των ψηφίων εισόδου N.

### Άσκηση 4: Πρόσθεση & αφαίρεση προσημασμένων και μη προσημασμένων αριθμών

Στο επόμενο σχήμα δείχνεται το διάγραμμα ενός αριθμητικού κυκλώματος που έχει την ακόλουθη διεπαφή:

**Είσοδοι:** **a (n-1:0)**, **b (n-1:0)**, τελεστές των n ψηφίων, **cin**, κρατούμενο εισόδου, **code (2:0)**, κωδικός λειτουργίας (βλέπε πίνακα).

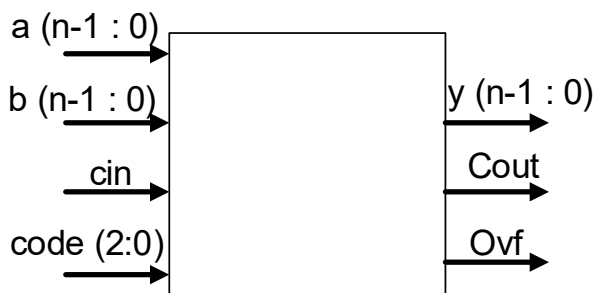
**Εξοδοι:** **y(n-1:0)** αποτέλεσμα υπολογισμού, **cout**, κρατούμενο εξόδου, **ovf** σημαία υπερχείλισης (overflow).

Το κύκλωμα επιτελεί τις πράξεις που φαίνονται στο διπλανό πίνακα.

Γράψτε κώδικα VHDL για το παραπάνω κύκλωμα, θεωρώντας ότι όλα τα σήματα εισόδου και εξόδου είναι τύπου `std_logic`. Ο κώδικας πρέπει να είναι παραμετρικός ως προς το πλήθος των ψηφίων εισόδου. Για το λόγο αυτό χρησιμοποιήστε την εντολή `GENERIC`

Για τις ακόλουθες εισόδους, προσομοιώστε τον κώδικα VHDL για όλες τις περιπτώσεις λειτουργίας (τιμές σήματος code). Είσοδος 1: (a, b, cin) = (0101, 0010,

1), Είσοδος 2: (a, b, cin) = (1000, 0011, 1), Είσοδος 3: (a, b, cin) = (1011, 1110, 1).  
 Εξηγήστε σύντομα αλλά πλήρως τα αποτελέσματα τις προσομοίωσης



Τύπος	Πράξη	Code
unsigned	$y = a + b$	000
	$y = a - b$	001
	$y = -a + b$	010
	$y = a + b + cin$	011
signed	$y = a + b$	100
	$y = a - b$	101
	$y = -a + b$	110
	$y = a + b + cin$	111

### Σημείωση

1. Όλες οι ασκήσεις να υλοποιηθούν με συντρέχων (concurrent) VHDL κώδικα. Δεν επιτρέπεται η χρήση port maps, process

2. Όλες τις ασκήσεις να συνοδεύονται από επαρκείς προσομοιώσεις οι οποίες πρέπει να σχολιάζονται ώστε να αποδεικνύεται ΠΛΗΡΩΣ η ορθότητα του κάθε κώδικα