



Customizing PSoC Designer™ User Modules

Doc. # 001-84020 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intl): 408.943.2600
<http://www.cypress.com>

Copyrights

© Cypress Semiconductor Corporation, 2012-2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

PSoC® is a registered trademark of Cypress Semiconductor Corporation. All products and company names mentioned in this document may be the trademarks of their respective holders.

Purchase of I2C components from Cypress or one of its sublicensed Associated Companies conveys a license under the Philips I2C Patent Rights to use these components in an I2C system, provided that the system conforms to the I2C Standard Specification as defined by Philips. As from October 1st, 2006 Philips Semiconductors has a new trade name - NXP Semiconductors.

Flash Code Protection

Cypress products meet the specifications contained in their particular Cypress PSoC Data Sheets. Cypress believes that its family of PSoC products is one of the most secure families of its kind on the market today, regardless of how they are used. There may be methods, unknown to Cypress, that can breach the code protection features. Any of these methods, to our knowledge, would be dishonest and possibly illegal. Neither Cypress nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Cypress is willing to work with the customer who is concerned about the integrity of their code. Code protection is constantly evolving. We at Cypress are committed to continuously improving the code protection features of our products.

Contents



1. Introduction	5
1.1 Overview	5
1.2 Acronyms Used in the Document	5
1.3 Documentation Conventions	5
1.4 Document Revision History	6
2. Creating a Custom User Module	7
2.1 Step 1: Copy and Name the User Module	7
2.1.1 API Register List	11
2.1.2 Modifying the UM	12
2.2 Step 2: Add Interrupt.....	12
2.3 Step 3: Remove Interrupt Type Parameter	14
2.4 Step 4: Add Interrupts to SHAPE.....	15
2.5 Step 5: Add Interrupt Handler	18
2.6 Step 6: Change APIs	19
2.7 Step 7: Edit the .inc file	20
2.8 Step 8: Edit the .h File	21
2.9 Step 9: Edit Placement Files.....	22
2.10 Verify All Modifications.....	23
2.11 Distribution	23
2.11.1 Exporting the User Module.....	23
2.11.2 Importing the User Module	24
3. Extending the Custom User Module	27
3.1 Add Wizard	28
3.1.1 Step 10: Create C# Package - General Rules	28
3.1.2 Step 11: Attach *.dll to UM	33
3.2 Verify Wizard Availability in Custom UM.....	35
3.2.1 Step 12: Update both xml and h files to manage time constant	35
3.2.2 Step 13: Implement Required Functionality in Wizard	38
3.3 Add Pre-Processing Javascript.....	41
3.3.1 Step 14: Create JS file	41
3.3.2 Step 15: Attach Pre-Processing Javascript to UM	41
3.3.3 Step 16: Implement Required Functionality in Pre-/Post-Processing JS	42
3.4 Add Placement Javascript	46
3.4.1 Step 17: Attach Placement Javascript to UM	46
3.4.2 Step #18: Implement Required Functionality in Placement JS	47
3.5 Add Code Generation Javascript.....	48
3.5.1 Step 19: Attach Code Generation Javascript to UM.....	48
3.5.2 Step 20: Implement Required Functionality in Code Generation JS	50
3.6 Add Design Rule Check Javascript	50
3.6.1 Step 21: Attach Design Rule Check Javascript to UM	50

3.6.2	Step 22: Implement Required Functionality in Design Rule Check JS	52
3.7	Add Un-placement Javascript.....	52
3.7.1	Step 23: Attach Un-placement Javascript to UM	52
3.7.2	Step 24: Implement Required Functionality in Un-placement JS.....	54
A.	Appendices	55
A.1	Appendix A	55
A.1.1	Timer16xPackage\Timer16xPackage\Timer16xPackage.cs	55
A.1.2	Timer16xPackage\Timer16xPackage\WizardForm.Designer.cs.....	57
A.1.3	Timer16xPackage\Timer16xPackage\WizardForm.cs	61
A.2	Appendix B	66
A.2.1	Timer16X\Ver_2_6\scripts\EventHandlers.js.....	66
A.3	Appendix C	68
A.3.1	Timer16X\Ver_2_6\scripts\Timer16xRules.js	68

1. Introduction



1.1 Overview

The objective of this guide is to create an improved user module (Timer16X), using the old user module (Timer16) as a template, with the help of the “User Module Customization Wizard” available in PSoC Designer™ 5.4. This guide describes how to modify an existing user module to meet your needs. The “User Module Customization” feature allows you to create a copy of an existing user module and export it.

Each user module is a combination of information on the interconnections of PSoC resources and the software used to control them. It is possible to generate new UMs or customize existing UMs. Different UMs can be combined to produce a new UM. These new UMs can be similar to the old ones, with no hardware changes and only with API changes.

1.2 Acronyms Used in the Document

Table 1-1. List of Acronyms

Acronym	Description
LSB	Least significant bit
MSB	Most significant bit
CRET	Close, Reopen with Explorer to Test
ISR	Interrupt service routine
API	Application Programming Interface

1.3 Documentation Conventions

Table 1-2. Document Conventions for Guides

Convention	Usage
Courier New	Displays file locations, user entered text, and source code: C:\ ...cd\icc\
<i>Italics</i>	Displays file names and reference documentation: Read about the <i>sourcefile.hex</i> file in the <i>PSoC Designer User Guide</i> .
[Bracketed, Bold]	Displays keyboard commands in procedures: [Enter] or [Ctrl] [C]

Table 1-2. Document Conventions for Guides (*continued*)

Convention	Usage
File > Open	Represents menu paths: File > Open > New Project
Bold	Displays commands, menu paths, and icon names in procedures: Click the File icon and then click Open .
Times New Roman	Displays an equation: $2 + 2 = 4$
Text in gray boxes	Describes cautions or unique functionality of the product.

1.4 Document Revision History

Table 1-3. Revision History

Revision	Date	Description of Change
**	12/21/2012	New specification.
*A	4/25/13	Added Extending section.

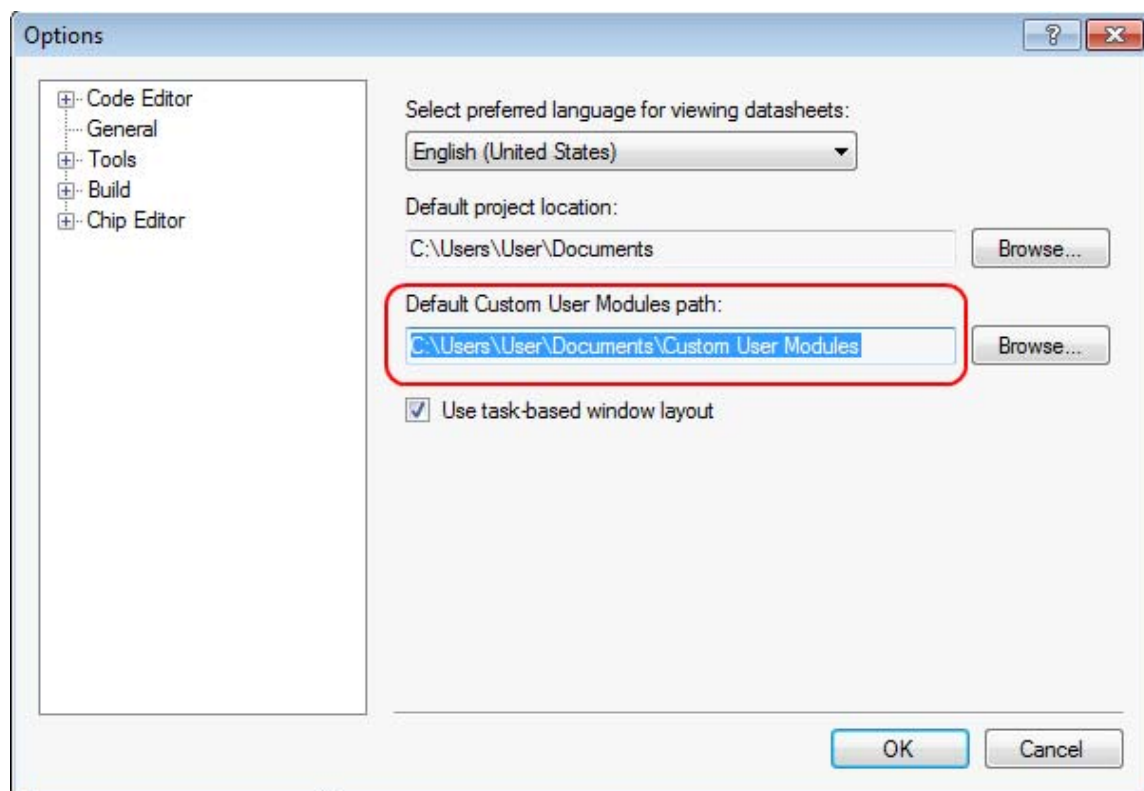
2. Creating a Custom User Module



The objective of this project is to build an improved Timer16 User Module, using the existing user module. The upper digital block of the present Timer16 has a parameter selection to generate interrupt on Capture, Terminal Count, and Compare True. The goal is to create a new UM (Timer16X) that generates a Terminal Count interrupt for the upper digital block and generates a Capture interrupt for the lower digital block.

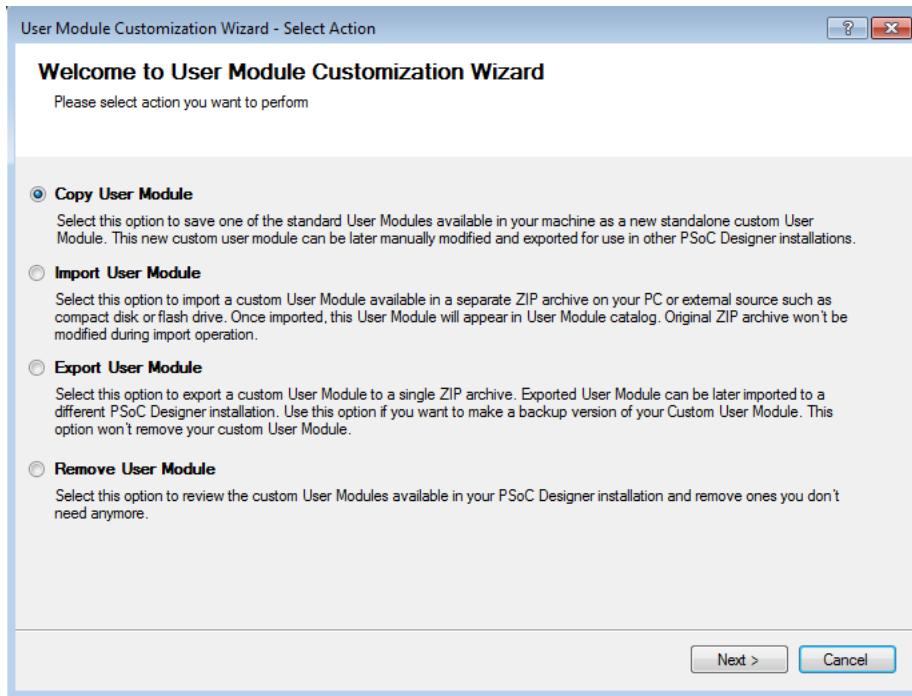
2.1 Step 1: Copy and Name the User Module

1. Launch PSoC Designer.
2. Select **Tools > Options > General** and set the default location in the 'Default Custom User Module path' tab to store and process custom UMs. There is only one folder with custom UMs in each PSoC Designer installation.

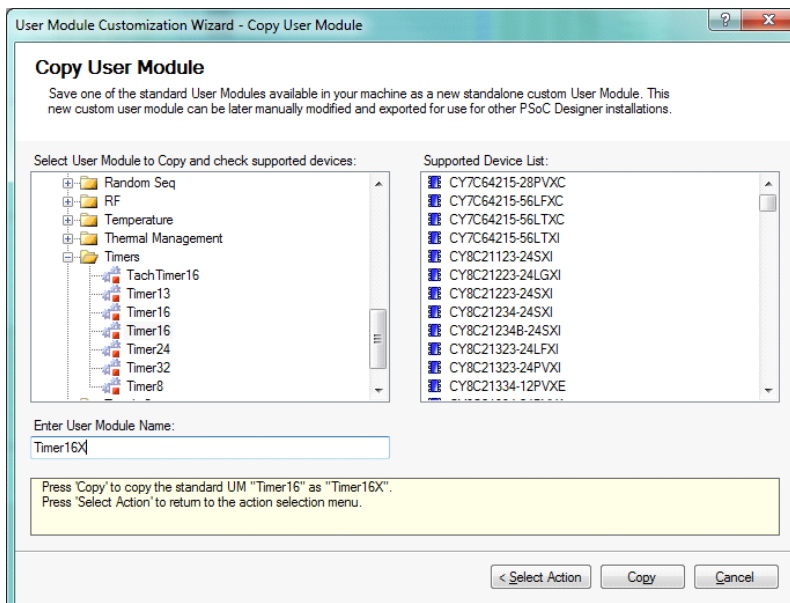


3. Go to **Tools > User Module Customization Wizard**.

4. Select the **Copy User Module** option and click **Next**.

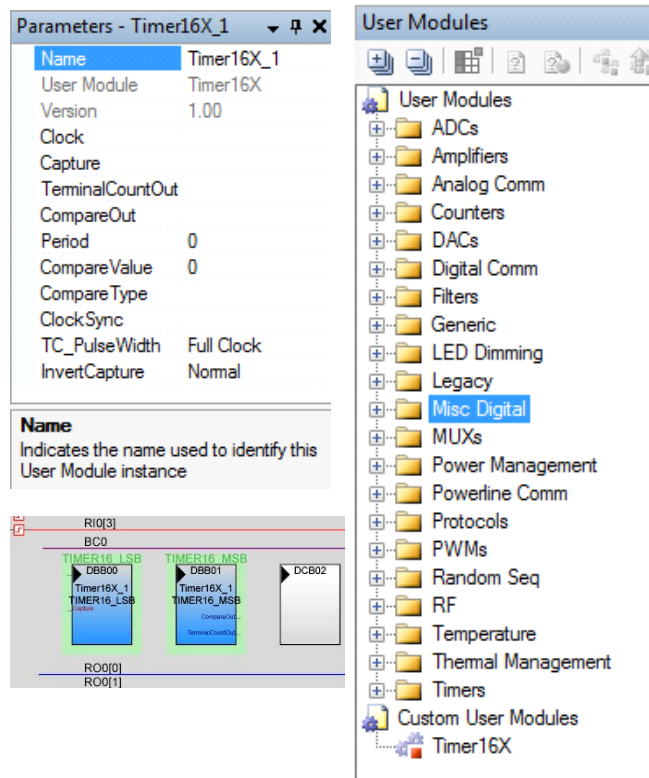


5. Choose 'Timer16' in the User Module tree.
6. Enter the name Timer16X.
7. Observe the list of devices that support the UM in the "Supported Device List" box.
8. Click **Copy** and then click **Close**.



9. Create the project based on the CY8C29666-24LFXI part (or any other part from the 'Supported Devices' List).
10. Verify that the UM is present and you are able to place it.

11. For UMs that have significant changes in API but no hardware changes, this is your chance to change the APIs to the way that you think they should be, while leaving the base UM hardware configuration unchanged.



The screenshot shows the PSoC Designer interface. On the left, the 'Parameters - Timer16X_1' window is open, displaying the following parameters:

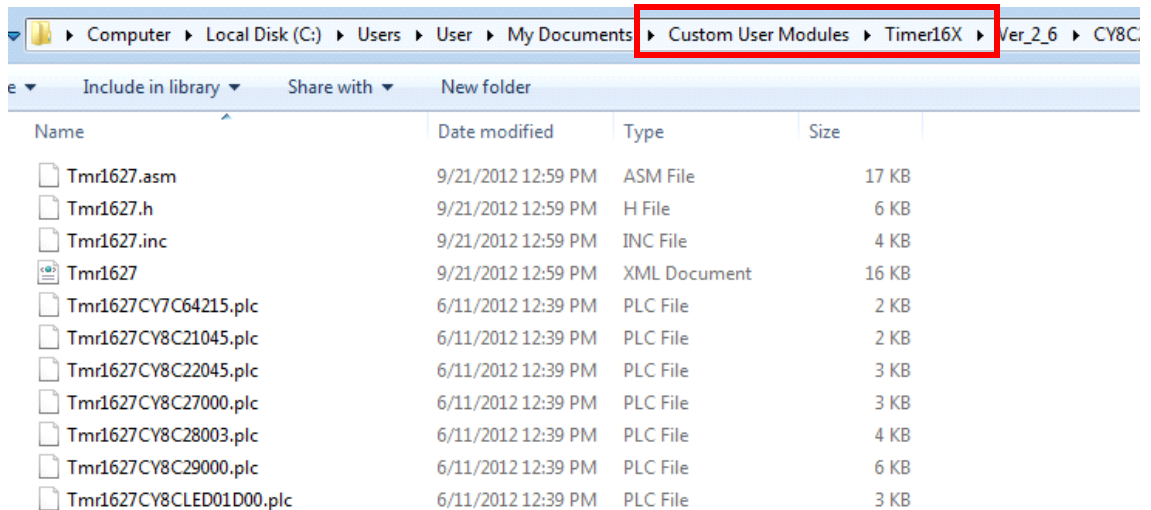
Name	Value
User Module	Timer16X
Version	1.00
Clock	
Capture	
TerminalCountOut	
CompareOut	
Period	0
CompareValue	0
CompareType	
ClockSync	
TC_PulseWidth	Full Clock
InvertCapture	Normal

Below the parameters, a note states: **Name** Indicates the name used to identify this User Module instance.

On the right, the 'User Modules' tree is visible, showing a hierarchy of modules. The 'Timer16X' module is highlighted under the 'Custom User Modules' folder.

12. Close PSoC Designer.

13. Go to the Timer16X folder in the folder with custom UMs.



The screenshot shows a Windows Explorer window with the following path: Computer > Local Disk (C:) > Users > User > My Documents > Custom User Modules > Timer16X > Ver_2_6 > CY8C27. The 'Timer16X' folder is highlighted in the address bar.

Name	Date modified	Type	Size
Tmr1627.asm	9/21/2012 12:59 PM	ASM File	17 KB
Tmr1627.h	9/21/2012 12:59 PM	H File	6 KB
Tmr1627.inc	9/21/2012 12:59 PM	INC File	4 KB
Tmr1627	9/21/2012 12:59 PM	XML Document	16 KB
Tmr1627CY7C64215.plc	6/11/2012 12:39 PM	PLC File	2 KB
Tmr1627CY8C21045.plc	6/11/2012 12:39 PM	PLC File	2 KB
Tmr1627CY8C22045.plc	6/11/2012 12:39 PM	PLC File	3 KB
Tmr1627CY8C27000.plc	6/11/2012 12:39 PM	PLC File	3 KB
Tmr1627CY8C28003.plc	6/11/2012 12:39 PM	PLC File	4 KB
Tmr1627CY8C29000.plc	6/11/2012 12:39 PM	PLC File	6 KB
Tmr1627CY8CLED01D00.plc	6/11/2012 12:39 PM	PLC File	3 KB

14. Open the CY8C27 UM sub-folder and open *the* Tmr1627.xml file with your preferred text editor

15. This file has four main sections:

- a. SHAPE

For regs/bit fields that are set and do not change.

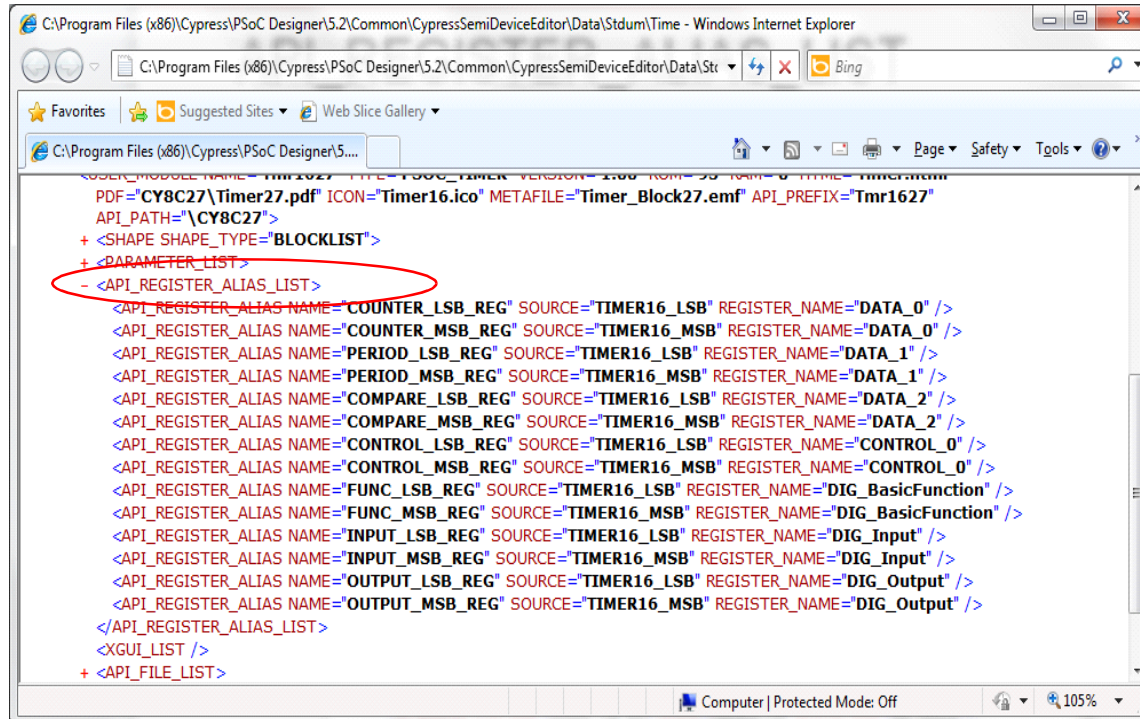
- b. **PARAMETER_LIST**
For bit fields that are set by user selection.
- c. **API_REGISTER_ALIAS_LIST**
Allows appropriate names of registers.
- d. **API_FILE LIST**
Lists the API files to be generated

```

1  <!--
2  ****
3  ****
4  **  FILENAME: Tmr1627.xml
5  **  Version: 2.6, Updated on 2012/10/30 at 20:40:30
6  **
7  **  DESCRIPTION: 16-bit Timer Personalization/Parameterization specificatio
8  **                  for the 22/24/27/28xxx PSoC family of devices
9  **  -----
10 **      Copyright (c) Cypress Semiconductor 2012. All Rights Reserved.
11 ****
12 ****
13 -->
14 <PSOC_DEVICE_DB>
15 <USER_MODULE_LIST>
16   <USER_MODULE NAME = "Tmr1627"
17     TYPE = "PSOC_TIMER"
18     VERSION = "2.6"
19     ROM = "93"
20     RAM = "0"
21     HTML = "Timer.html"
22     PDF = "CY8C27\Timer27.pdf"
23     ICON = "Timer16.ico"
24     METAFILE = "Timer_Block27.emf"
25     API_PREFIX = "Tmr1627"
26     API_PATH = "\CY8C27"
27   >
28   <SHAPE SHAPE_TYPE = "BLOCKLIST">
29     <BLOCK_LIST>
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65   </SHAPE>
66
67   <PARAMETER_LIST>
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2
```

2.1.1 API Register List

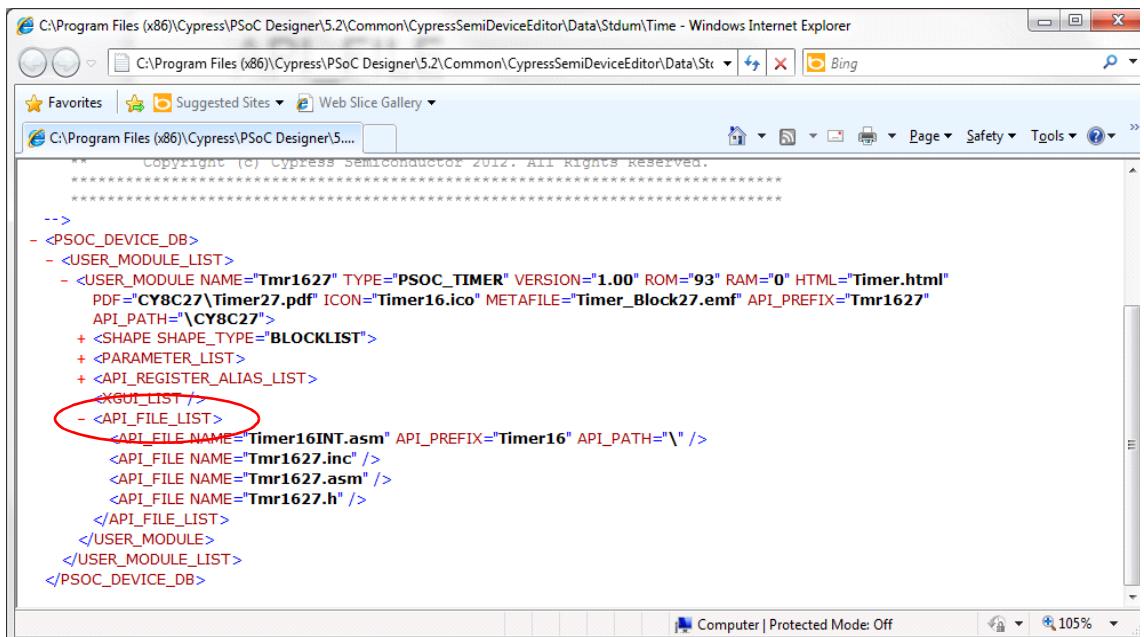
- The API_REGISTER_ALIAS_LIST allows you to define register names.



```

C:\Program Files (x86)\Cypress\PSoC Designer\5.2\Common\CypressSemiDeviceEditor\Data\Stdum\Time - Windows Internet Explorer
C:\Program Files (x86)\Cypress\PSoC Designer\5.2\Common\CypressSemiDeviceEditor\Data\Stdum\Time - Windows Internet Explorer
PDF=>CY8C27\Timer27.pdf ICON=>Timer16.ico METAFILE=>Timer_Block27.emf API_PREFIX=>Tmr1627
API_PATH=>\CY8C27>
+ <SHAPE SHAPE_TYPE="BLOCKLIST">
+ <PARAMETER_LIST>
- <API_REGISTER_ALIAS_LIST>
  <API_REGISTER_ALIAS NAME="COUNTER_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="DATA_0" />
  <API_REGISTER_ALIAS NAME="COUNTER_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="DATA_0" />
  <API_REGISTER_ALIAS NAME="PERIOD_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="DATA_1" />
  <API_REGISTER_ALIAS NAME="PERIOD_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="DATA_1" />
  <API_REGISTER_ALIAS NAME="COMPARE_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="DATA_2" />
  <API_REGISTER_ALIAS NAME="COMPARE_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="DATA_2" />
  <API_REGISTER_ALIAS NAME="CONTROL_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="CONTROL_0" />
  <API_REGISTER_ALIAS NAME="CONTROL_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="CONTROL_0" />
  <API_REGISTER_ALIAS NAME="FUNC_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="DIG_BasicFunction" />
  <API_REGISTER_ALIAS NAME="FUNC_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="DIG_BasicFunction" />
  <API_REGISTER_ALIAS NAME="INPUT_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="DIG_Input" />
  <API_REGISTER_ALIAS NAME="INPUT_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="DIG_Input" />
  <API_REGISTER_ALIAS NAME="OUTPUT_LSB_REG" SOURCE="TIMER16_LSB" REGISTER_NAME="DIG_Output" />
  <API_REGISTER_ALIAS NAME="OUTPUT_MSB_REG" SOURCE="TIMER16_MSB" REGISTER_NAME="DIG_Output" />
</API_REGISTER_ALIAS_LIST>
<XGUI_LIST />
+ <API_FILE_LIST>
  
```

- The API_FILE_LIST allows you to define the name and location of all API files. It is possible to add multiple assembly files and C routines.



```

C:\Program Files (x86)\Cypress\PSoC Designer\5.2\Common\CypressSemiDeviceEditor\Data\Stdum\Time - Windows Internet Explorer
C:\Program Files (x86)\Cypress\PSoC Designer\5.2\Common\CypressSemiDeviceEditor\Data\Stdum\Time - Windows Internet Explorer
Copyright (c) Cypress Semiconductor 2012. All Rights Reserved.
*****
-->
- <PSOC_DEVICE_DB>
- <USER_MODULE_LIST>
  - <USER_MODULE NAME="Tmr1627" TYPE="PSOC_TIMER" VERSION="1.00" ROM="93" RAM="0" HTML="Timer.html"
    PDF=>CY8C27\Timer27.pdf ICON=>Timer16.ico METAFILE=>Timer_Block27.emf API_PREFIX=>Tmr1627
    API_PATH=>\CY8C27>
    + <SHAPE SHAPE_TYPE="BLOCKLIST">
    + <PARAMETER_LIST>
    + <API_REGISTER_ALIAS_LIST>
    <XGUI_LIST />
  - <API_FILE_LIST>
    <API_FILE NAME="Timer16INT.asm" API_PREFIX="Timer16" API_PATH="" />
    <API_FILE NAME="Tmr1627.inc" />
    <API_FILE NAME="Tmr1627.asm" />
    <API_FILE NAME="Tmr1627.h" />
  </API_FILE_LIST>
</USER_MODULE>
</USER_MODULE_LIST>
</PSOC_DEVICE_DB>
  
```

2.1.2 Modifying the UM

To modify this UM, the LSB block must be set to be an interrupt source in the SHAPE section. The parameter used to select the interrupt type must be removed from the PARAMETER_LIST area. Bit values must be added to the SHAPE section to set the interrupt type to the following:

- Terminal Count for the MSB Block
- Capture for the LSB Block

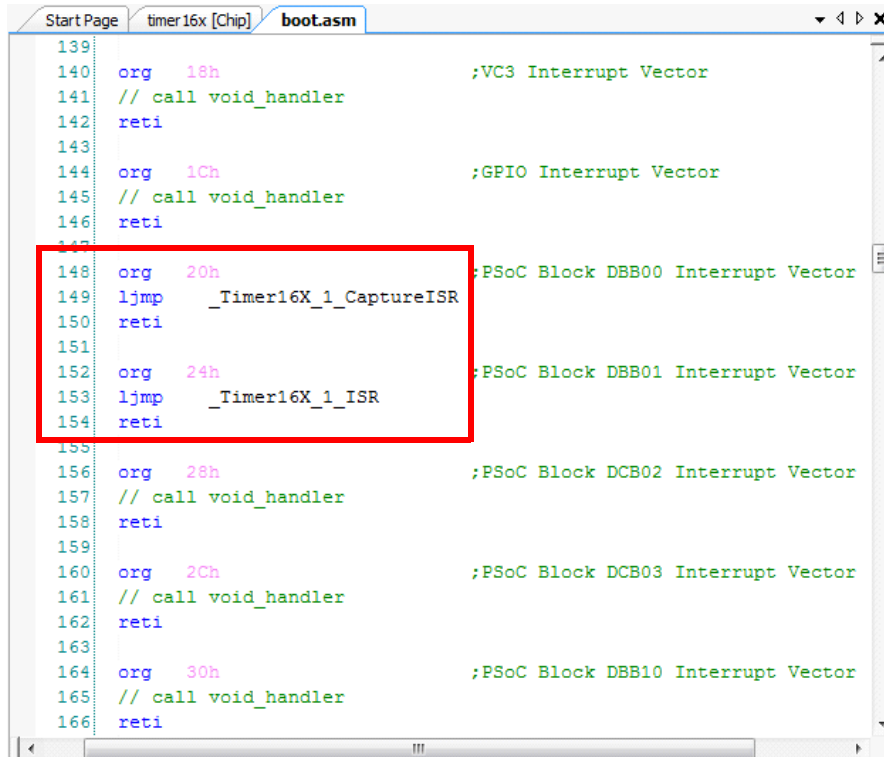
2.2 Step 2: Add Interrupt

1. Open *Tmr1627.xml*.
2. Use the interrupt code from the TIMER16_MSB block to add an interrupt to the Timer16_LSB block. Name this new interrupt source `_CaptureISR`.
3. Close the file and reopen with Internet Explorer. This checking process is done frequently; you can name it 'Close, Reopen with Explorer to Test' (CRET).



4. Open a CY8C29666-24LFXI part-based project (or any other part from the Supported Devices List) with PSoC Designer.

5. Place Timer16X.
6. Generate the application.
7. Go to the application editor.
8. Open *boot.asm*.
9. Verify that the new interrupt vector has been added.



```

139
140 org 18h ;VC3 Interrupt Vector
141 // call void_handler
142 reti
143
144 org 1Ch ;GPIO Interrupt Vector
145 // call void_handler
146 reti
147
148 org 20h ;PSoC Block DBB00 Interrupt Vector
149 ljmp _Timer16X_1_CaptureISR
150 reti
151
152 org 24h ;PSoC Block DBB01 Interrupt Vector
153 ljmp _Timer16X_1_ISR
154 reti
155
156 org 28h ;PSoC Block DCB02 Interrupt Vector
157 // call void_handler
158 reti
159
160 org 2Ch ;PSoC Block DCB03 Interrupt Vector
161 // call void_handler
162 reti
163
164 org 30h ;PSoC Block DBB10 Interrupt Vector
165 // call void_handler
166 reti
  
```

2.3 Step 3: Remove Interrupt Type Parameter

1. Close PSoC Designer.
2. Open Tmr1627.xml.
3. Remove the InterruptType parameter.
4. Save the removed portion in a temporary file.
5. For the ClockSync parameter, change the ORDER attribute from 8 to 7.
6. For the TC_Width parameter, change the ORDER attribute from 9 to 8.

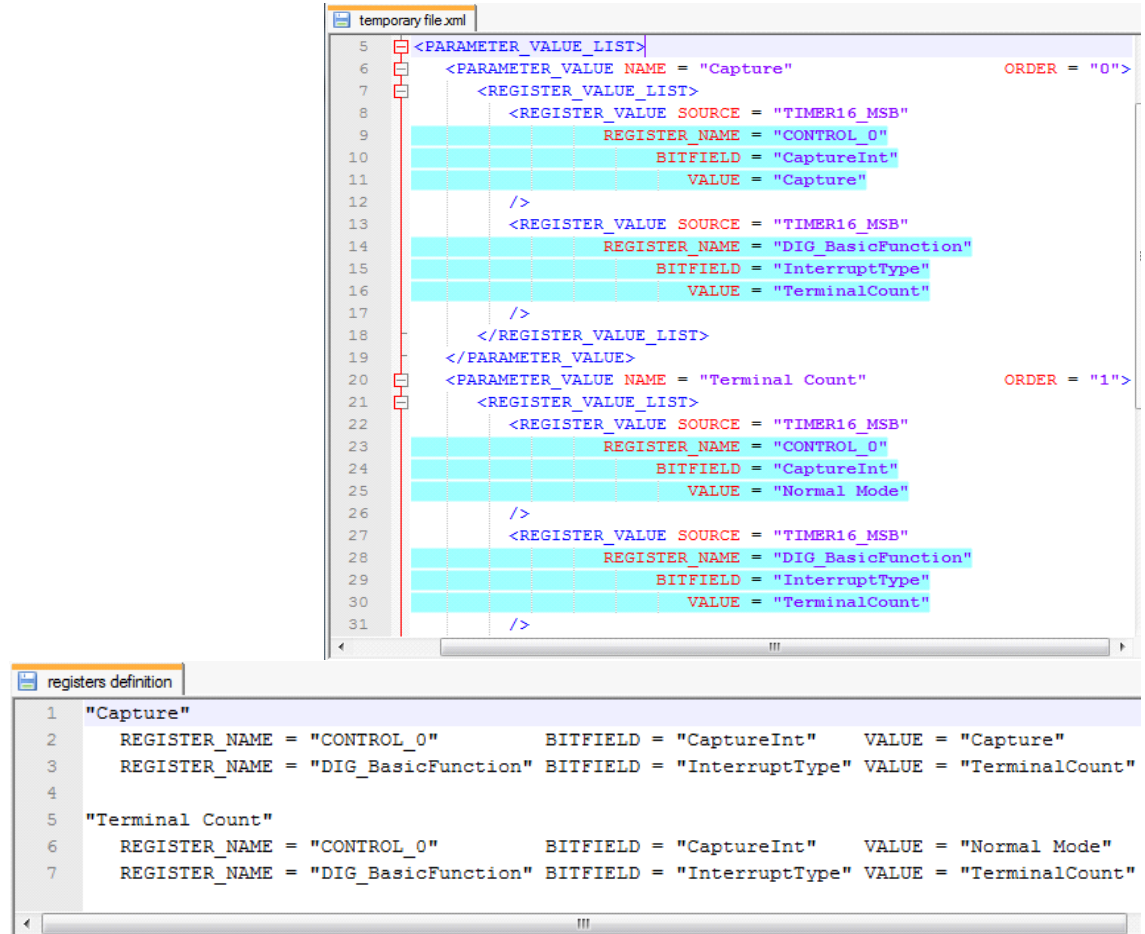


```

170     </PARAMETER_VALUE>
171   </PARAMETER_VALUE_LIST>
172 </PARAMETER>
173 <PARAMETER NAME = "InterruptType"
174   ORDER = "7"
175   TYPE = "BLOCK"
176 >
177   <PARAMETER_VALUE_LIST>
178     <PARAMETER_VALUE NAME = "Capture" ORDER = "0">
179       <REGISTER_VALUE_LIST>
180         <REGISTER_VALUE SOURCE = "TIMER16_MSB"
181           REGISTER_NAME = "CONTROL_0"
182           BITFIELD = "CaptureInt"
183           VALUE = "Capture"
184         />
185         <REGISTER_VALUE SOURCE = "TIMER16_MSB"
186           REGISTER_NAME = "DIG_BasicFunction"
187           BITFIELD = "InterruptType"
188           VALUE = "TerminalCount"
189         />
190       </REGISTER_VALUE_LIST>
191     </PARAMETER_VALUE>
192     <PARAMETER_VALUE NAME = "Terminal Count" ORDER = "1">
193       <REGISTER_VALUE_LIST>
194         <REGISTER_VALUE SOURCE = "TIMER16_MSB"
195           REGISTER_NAME = "CONTROL_0"
196           BITFIELD = "CaptureInt"
197           VALUE = "Normal Mode"
198         />
199         <REGISTER_VALUE SOURCE = "TIMER16_MSB"
200           REGISTER_NAME = "DIG_BasicFunction"
201           BITFIELD = "InterruptType"
202           VALUE = "TerminalCount"
203         />
204       </REGISTER_VALUE_LIST>
205     </PARAMETER_VALUE>
206     <PARAMETER_VALUE NAME = "Compare True" ORDER = "2">
207       <REGISTER_VALUE_LIST>
208         <REGISTER_VALUE SOURCE = "TIMER16_MSB"
209           REGISTER_NAME = "CONTROL_0"
210           BITFIELD = "CaptureInt"
211           VALUE = "Normal Mode"
212         />
213         <REGISTER_VALUE SOURCE = "TIMER16_MSB"
214           REGISTER_NAME = "DIG_BasicFunction"
215           BITFIELD = "InterruptType"
216           VALUE = "CompareTrue"
217         />
218       </REGISTER_VALUE_LIST>
219     </PARAMETER_VALUE>
220   </PARAMETER_VALUE_LIST>
221 </PARAMETER>
222 <PARAMETER NAME = "ClockSync"
223   ORDER = "7"
  
```

2.4 Step 4: Add Interrupts to SHAPE

1. Open the temporary file where you pasted the InterruptType parameter.
2. The REGISTER_NAME, BITFIELD and VALUE attributes for each type of interrupt can be used to define the required registers.



The image shows two XML files. The top file, 'temporary file.xml', contains two parameter value lists. The first list, for 'Capture', defines two registers: 'CONTROL_0' with bitfield 'CaptureInt' and value 'Capture', and 'DIG_BasicFunction' with bitfield 'InterruptType' and value 'TerminalCount'. The second list, for 'Terminal Count', defines two registers: 'CONTROL_0' with bitfield 'CaptureInt' and value 'Normal Mode', and 'DIG_BasicFunction' with bitfield 'InterruptType' and value 'TerminalCount'.

The bottom file, 'registers definition', shows the resulting register definitions for the 'Capture' and 'Terminal Count' parameters. For 'Capture', it defines 'CONTROL_0' with bitfield 'CaptureInt' and value 'Capture', and 'DIG_BasicFunction' with bitfield 'InterruptType' and value 'TerminalCount'. For 'Terminal Count', it defines 'CONTROL_0' with bitfield 'CaptureInt' and value 'Normal Mode', and 'DIG_BasicFunction' with bitfield 'InterruptType' and value 'TerminalCount'.

3. Open Tmr1627.xml.
4. For TIMER16_LSB, add "InterruptType" bit field to the "DIG_BasicFunction" register. Set its value to "TerminalCount".

5. Add the CONTROL_0 register. Add the "CaptureInt" bit field to it. Set its value to "Capture".

```

28 <SHAPE SHAPE_TYPE = "BLOCKLIST">
29   <BLOCK_LIST>
30     <BLOCK
31       NAME = "TIMER16_LSB"
32       TYPE = "DIGITAL"
33       INTERRUPT_SOURCE = "_CaptureISR"
34       INTERRUPT_TYPE = "JUMP"
35     >
36     <REGISTER_LIST>
37       <REGISTER NAME = "DIG_BasicFunction">
38         <BITFIELD_LIST>
39           <BITFIELD NAME = "Function" VALUE = "Timer" />
40           <BITFIELD NAME = "End" VALUE = "NotEnd" />
41           <BITFIELD NAME = "InterruptType" VALUE = "TerminalCount" />
42         </BITFIELD_LIST>
43       </REGISTER>
44       <REGISTER NAME = "CONTROL_0">
45         <BITFIELD_LIST>
46           <BITFIELD NAME = "CaptureInt" VALUE = "Capture" />
47         </BITFIELD_LIST>
48       </REGISTER>
49     </REGISTER_LIST>
50     <INPUT_LIST/>
51   </BLOCK>
52   <BLOCK
53     NAME = "TIMER16_MSB"
54     TYPE = "DIGITAL"
55     INTERRUPT_SOURCE = "_ISR"
56     INTERRUPT_TYPE = "JUMP"
57   >
58   <REGISTER_LIST>
59     <REGISTER NAME = "DIG_BasicFunction">
60       <BITFIELD_LIST>
61         <BITFIELD NAME = "Function" VALUE = "Timer" />
62         <BITFIELD NAME = "End" VALUE = "IsEnd" />
63       </BITFIELD_LIST>
64     </REGISTER>
65   </REGISTER_LIST>
66   <INPUT_LIST>
67     <INPUT
68       SOURCE = "TIMER16_LSB"
69       TYPE = "BLOCK"
70       REGISTER_NAME = "DIG_Input"
71       BITFIELD = "DataSelect"
72     />
73   </INPUT_LIST>
74 </BLOCK>
75 </BLOCK_LIST>
76 </SHAPE>

```

6. For TIMER16_MSB, add "InterruptType" bit field to the "DIG_BasicFunction" register. Set its value to "TerminalCount".

7. Add the CONTROL_0 register. Add the "CaptureInt" bit field to it. Set its value to "Normal Mode".

```

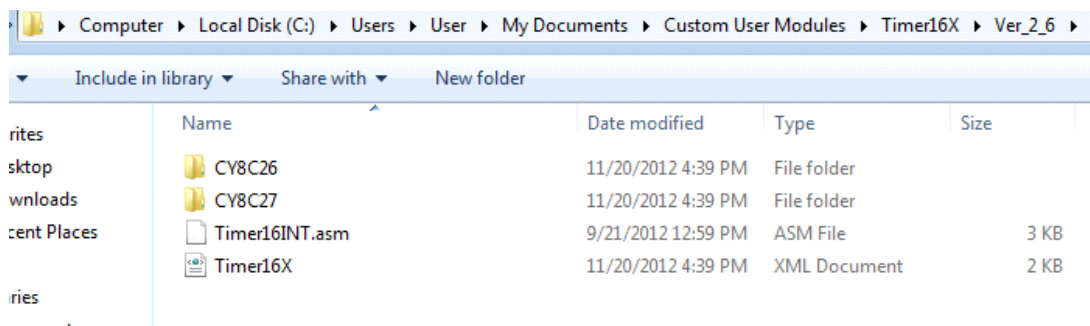
28 <SHAPE SHAPE_TYPE = "BLOCKLIST">
29   <BLOCK_LIST>
30     <BLOCK
31       NAME = "TIMER16_LSB"
32       TYPE = "DIGITAL"
33       INTERRUPT_SOURCE = "_CaptureISR"
34       INTERRUPT_TYPE = "JUMP"
35     >
36     <REGISTER_LIST>
37       <REGISTER NAME = "DIG_BasicFunction">
38         <BITFIELD_LIST>
39           <BITFIELD NAME = "Function" VALUE = "Timer" />
40           <BITFIELD NAME = "End" VALUE = "NotEnd" />
41           <BITFIELD NAME = "InterruptType" VALUE = "TerminalCount" />
42         </BITFIELD_LIST>
43       </REGISTER>
44       <REGISTER NAME = "CONTROL_0">
45         <BITFIELD_LIST>
46           <BITFIELD NAME = "CaptureInt" VALUE = "Capture" />
47         </BITFIELD_LIST>
48       </REGISTER>
49     </REGISTER_LIST>
50     <INPUT_LIST/>
51   </BLOCK>
52   <BLOCK
53     NAME = "TIMER16_MSB"
54     TYPE = "DIGITAL"
55     INTERRUPT_SOURCE = "_ISR"
56     INTERRUPT_TYPE = "JUMP"
57   >
58   <REGISTER_LIST>
59     <REGISTER NAME = "DIG_BasicFunction">
60       <BITFIELD_LIST>
61         <BITFIELD NAME = "Function" VALUE = "Timer" />
62         <BITFIELD NAME = "End" VALUE = "IsEnd" />
63         <BITFIELD NAME = "InterruptType" VALUE = "TerminalCount" />
64       </BITFIELD_LIST>
65     </REGISTER>
66     <REGISTER NAME = "CONTROL_0">
67       <BITFIELD_LIST>
68         <BITFIELD NAME = "CaptureInt" VALUE = "Normal Mode" />
69       </BITFIELD_LIST>
70     </REGISTER>
71   </REGISTER_LIST>
72   <INPUT_LIST>
73     <INPUT
74       SOURCE = "TIMER16_LSB"
75       TYPE = "BLOCK"
76       REGISTER_NAME = "DIG_Input"
77       BITFIELD = "DataSelect"
78     />
79   </INPUT_LIST>
80 </BLOCK>
81 </BLOCK_LIST>
82 </SHAPE>
  
```

To modify firmware for this UM, you must do the following:

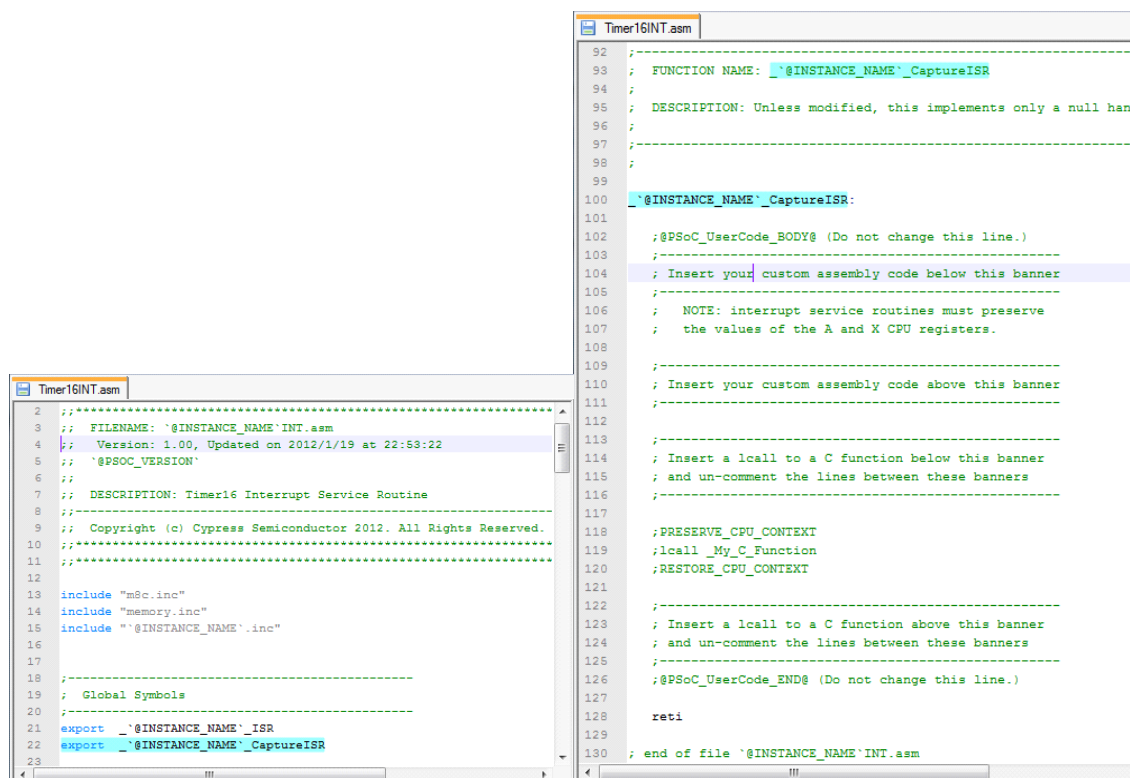
- Add a new interrupt handler for the new interrupt.
- Change the API to pass the intended interrupt to enable or disable type.
- Generate a new interrupt mask for this interrupt.
- For both C (.h) and assembly (.inc), change the placement file for the CY8C29000 parts so that both the interrupt masks are always in the same interrupt register.

2.5 Step 5: Add Interrupt Handler

1. Go to the Timer16X folder in the folder with custom UMs. Go to Timer16INT.asm file, which is located on the same level as the CY8C27 folder and open it.



2. Create a duplicate _ISR function and rename it as _CaptureISR.
3. Add 'export' to the new label and save the file.



2.6 Step 6: Change APIs

1. Open the CY8C27 folder and open Tmr1627.asm.
2. Both `@INSTANCE_NAME`_EnableInt and `@INSTANCE_NAME`_DisableInt APIs are macros. These macros are defined in Tmr1627.inc.
3. Although not needed for this particular example, this step is shown because many UMs you develop will require some assembly code alteration.

```

Timer16INT.asm  Tmr1627.asm
77 .SECTION
78 ;-----
79 ; FUNCTION NAME: `@INSTANCE_NAME`_EnableInt
80 ;
81 ; DESCRIPTION:
82 ;   Enables this timer's interrupt by setting the interrupt enable mask bit
83 ;   associated with this User Module. This function has no effect until and
84 ;   unless the global interrupts are enabled (for example by using the
85 ;   macro MSC_EnableGInt).
86 ;-----
87 ;
88 ; ARGUMENTS:   None.
89 ; RETURNS:    Nothing.
90 ; SIDE EFFECTS:
91 ;   The A and X registers may be modified by this or future implementations
92 ;   of this function. The same is true for all RAM page pointer registers in
93 ;   the Large Memory Model. When necessary, it is the calling function's
94 ;   responsibility to preserve their values across calls to fastcall16
95 ;   functions.
96 ;
97 `@INSTANCE_NAME`_EnableInt:
98 `@INSTANCE_NAME`_EnableInt:
99   RAM_PROLOGUE RAM_USE_CLASS_1
100   `@INSTANCE_NAME`_EnableInt_M
101   RAM_EPILOGUE RAM_USE_CLASS_1
102   ret
103
104 .ENDSECTION

```

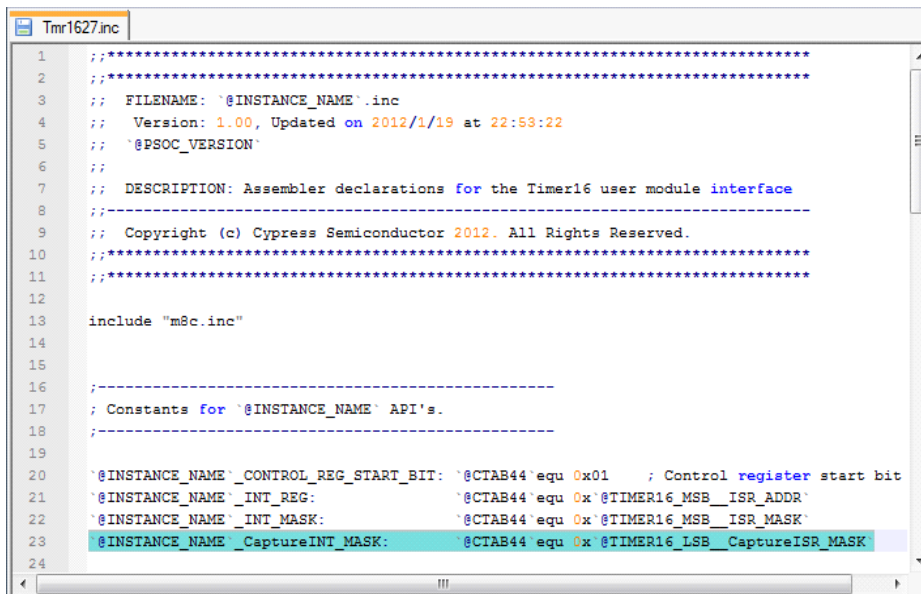
```

Timer16INT.asm  Tmr1627.asm
107 .SECTION
108 ;-----
109 ; FUNCTION NAME: `@INSTANCE_NAME`_DisableInt
110 ;
111 ; DESCRIPTION:
112 ;   Disables this timer's interrupt by clearing the interrupt enable
113 ;   mask bit associated with this User Module.
114 ;-----
115 ;
116 ; ARGUMENTS:   None
117 ; RETURNS:    Nothing
118 ; SIDE EFFECTS:
119 ;   The A and X registers may be modified by this or future implementations
120 ;   of this function. The same is true for all RAM page pointer registers in
121 ;   the Large Memory Model. When necessary, it is the calling function's
122 ;   responsibility to preserve their values across calls to fastcall16
123 ;   functions.
124 ;
125 `@INSTANCE_NAME`_DisableInt:
126 `@INSTANCE_NAME`_DisableInt:
127   RAM_PROLOGUE RAM_USE_CLASS_1
128   `@INSTANCE_NAME`_DisableInt_M
129   RAM_EPILOGUE RAM_USE_CLASS_1
130   ret
131
132 .ENDSECTION

```

2.7 Step 7: Edit the .inc file

1. Open *Tmr1627.inc*.
2. Duplicate the `_INT_ MASK` declaration and change for new `_CaptureINT_MASK` declaration.
3. Change enable and disable interrupt macros to use 'A' as the enabling and disabling mask.



```

1  ;;*****
2  ;;*****
3  ;;  FILENAME: `@INSTANCE_NAME`.inc
4  ;;  Version: 1.00, Updated on 2012/1/19 at 22:53:22
5  ;;  `@PSOC_VERSION`
6  ;;
7  ;;  DESCRIPTION: Assembler declarations for the Timer16 user module interface
8  ;;*****
9  ;;  Copyright (c) Cypress Semiconductor 2012. All Rights Reserved.
10 ;;*****
11 ;;*****
12
13 include "m8c.inc"
14
15
16 ;-----
17 ; Constants for `@INSTANCE_NAME` API's.
18 ;-----
19
20 `@INSTANCE_NAME`_CONTROL_REG_START_BIT: `@CTAB44`equ 0x01    ; Control register start bit
21 `@INSTANCE_NAME`_INT_REG:              `@CTAB44`equ 0x`@TIMER16_MSB__ISR_ADDR`
22 `@INSTANCE_NAME`_INT_MASK:              `@CTAB44`equ 0x`@TIMER16_MSB__ISR_MASK`
23 `@INSTANCE_NAME`_CaptureINT_MASK:      `@CTAB44`equ 0x`@TIMER16_LSB__CaptureISR_MASK`
24

```

```

65 macro `@INSTANCE_NAME`_EnableInt_M
66 ;M8C_EnableIntMask `@INSTANCE_NAME`_INT_REG, `@INSTANCE_NAME`_INT_MASK
67 mov X, SP
68 push A
69 mov A, reg[`@INSTANCE_NAME`_INT_REG]
70 or A, [X]
71 mov reg[`@INSTANCE_NAME`_INT_REG], A
72 pop A
73 endm
74
75 macro `@INSTANCE_NAME`_DisableInt_M
76 ;M8C_DisableIntMask `@INSTANCE_NAME`_INT_REG, `@INSTANCE_NAME`_INT_MASK
77 mov X, SP
78 cpl A
79 push A
80 mov A, reg[`@INSTANCE_NAME`_INT_REG]
81 and A, [X]
82 mov reg[`@INSTANCE_NAME`_INT_REG], A
83 pop A
84 endm

```

2.8 Step 8: Edit the .h File

1. Open *Tmr1627.h*.
2. Change the EnableInt and DisableInt function prototypes to require a BYTE argument.
3. Duplicate the _INT_ MASK #define and change the new _CaptureINT_MASK define.

```

37 //-----
38 // Prototypes of the `@INSTANCE_NAME` API.
39 //-----
40
41 //extern void `@INSTANCE_NAME`_EnableInt(void); // Proxy 1
42 extern void `@INSTANCE_NAME`_EnableInt(BYTE bMask); // Proxy 1
43 //extern void `@INSTANCE_NAME`_DisableInt(void); // Proxy 1
44 extern void `@INSTANCE_NAME`_DisableInt(BYTE bMask); // Proxy 1
45 extern void `@INSTANCE_NAME`_Start(void); // Proxy 1
46 extern void `@INSTANCE_NAME`_Stop(void); // Proxy 1
47 extern WORD `@INSTANCE_NAME`_wReadTimer(void); // Proxy 1
48 extern WORD `@INSTANCE_NAME`_wReadTimerSaveCV(void); // Proxy 2
49 extern void `@INSTANCE_NAME`_WritePeriod(WORD wPeriod); // Proxy 1
50 extern WORD `@INSTANCE_NAME`_wReadCompareValue(void); // Proxy 1
51 extern void `@INSTANCE_NAME`_WriteCompareValue(WORD wCompareValue); // Proxy 1
52
53 // The following functions are deprecated.
54 // They may be omitted in future releases
55 //
56 extern WORD w`@INSTANCE_NAME`_ReadCompareValue(void); // Deprecated
57 extern WORD w`@INSTANCE_NAME`_ReadTimerSaveCV(void); // Deprecated
58 extern WORD w`@INSTANCE_NAME`_ReadCounter(void); // Obsolete
59 extern WORD w`@INSTANCE_NAME`_ReadTimer(void); // Deprecated
60 extern WORD w`@INSTANCE_NAME`_CaptureCounter(void); // Obsolete
61
62
63 //-----
64 // Constants for `@INSTANCE_NAME` API's.
65 //-----
66
67 #define `@INSTANCE_NAME`_CONTROL_REG_START_BIT `@CTAB48` ( 0x01 )
68 #define `@INSTANCE_NAME`_INT_REG_ADDR `@CTAB48` ( 0x`@TIMER16_MSB_ISR_ADDR` )
69 #define `@INSTANCE_NAME`_INT_MASK `@CTAB48` ( 0x`@TIMER16_MSB_ISR_MASK` )
70 #define `@INSTANCE_NAME`_CaptureINT_MASK `@CTAB48` ( 0x`@TIMER16_LSB_CaptureISR_MASK` )
71

```

2.9 Step 9: Edit Placement Files

Placement files list all legal placements for a particular chip family. Only the CY829000 family allows placement so that each block has a different interrupt register for the DCB13 DBB20 blocks.

Open *Tmr1627CY8C2900.plc*. Remove RESOURCE_PLACEMENT_INDEX="7" and renumber all the following indices (highlighted by a red box in the following graphic).



```

44 <RESOURCE_PLACEMENT_LIST>
45 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DCB12"/>
46 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DCB13"/>
47 </RESOURCE_PLACEMENT_LIST>
48 </RESOURCE_LOCATION>
49 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="7">
50 <RESOURCE_PLACEMENT_LIST>
51 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DCB13"/>
52 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DBB20"/>
53 </RESOURCE_PLACEMENT_LIST>
54 </RESOURCE_LOCATION>
55 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="7">
56 <RESOURCE_PLACEMENT_LIST>
57 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DBB20"/>
58 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DBB21"/>
59 </RESOURCE_PLACEMENT_LIST>
60 </RESOURCE_LOCATION>
61 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="8">
62 <RESOURCE_PLACEMENT_LIST>
63 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DBB21"/>
64 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DCB22"/>
65 </RESOURCE_PLACEMENT_LIST>
66 </RESOURCE_LOCATION>
67 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="9">
68 <RESOURCE_PLACEMENT_LIST>
69 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DCB22"/>
70 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DCB23"/>
71 </RESOURCE_PLACEMENT_LIST>
72 </RESOURCE_LOCATION>
73 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="10">
74 <RESOURCE_PLACEMENT_LIST>
75 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DCB23"/>
76 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DBB30"/>
77 </RESOURCE_PLACEMENT_LIST>
78 </RESOURCE_LOCATION>
79 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="11">
80 <RESOURCE_PLACEMENT_LIST>
81 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DBB30"/>
82 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DBB31"/>
83 </RESOURCE_PLACEMENT_LIST>
84 </RESOURCE_LOCATION>
85 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="12">
86 <RESOURCE_PLACEMENT_LIST>
87 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DBB31"/>
88 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DCB32"/>
89 </RESOURCE_PLACEMENT_LIST>
90 </RESOURCE_LOCATION>
91 <RESOURCE_LOCATION RESOURCE_PLACEMENT_INDEX="13">
92 <RESOURCE_PLACEMENT_LIST>
93 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_LSB" TARGET_RESOURCE_NAME="DCB32"/>
94 <RESOURCE_PLACEMENT UM_RESOURCE_NAME="TIMER16_MSB" TARGET_RESOURCE_NAME="DCB33"/>
95 </RESOURCE_PLACEMENT_LIST>
96 </RESOURCE_LOCATION>
97 </RESOURCE_LOCATION_LIST>

```

2.10 Verify All Modifications

1. Start a PSoC Designer project for the CY8C29666-24LFXI part.
2. Verify the correctness of all placement combinations, which are represented in the modified *Tmr1627CY8C2900.plc* placement file. The following is a list of the Timer16X legal placements:

#	TIMER16_LSB	TIMER16_MSB
1	DBB00	DBB01
2	DBB01	DCB02
3	DCB02	DCB03
4	DCB03	DBB10
5	DBB10	DBB11
6	DBB11	DCB12
7	DCB12	DCB13
8	DBB20	DBB21
9	DBB21	DCB22
10	DCB22	DCB23
11	DCB23	DBB30
12	DBB30	DBB31
13	DBB31	DCB32
14	DCB32	DCB33

3. Generate the application. Verify that the API compiles correctly.

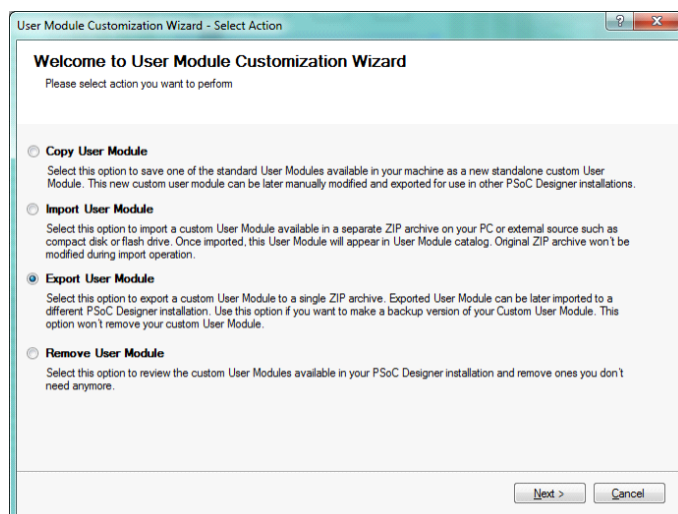
2.11 Distribution

After the new user module is code complete and tested, you can export Timer16X User Module to a single zip archive. The exported user module can be imported to a different PSoC Designer installation on another machine.

2.11.1 Exporting the User Module

1. Launch PSoC Designer.
2. Go to: **Tools > User Module Customization Wizard**.

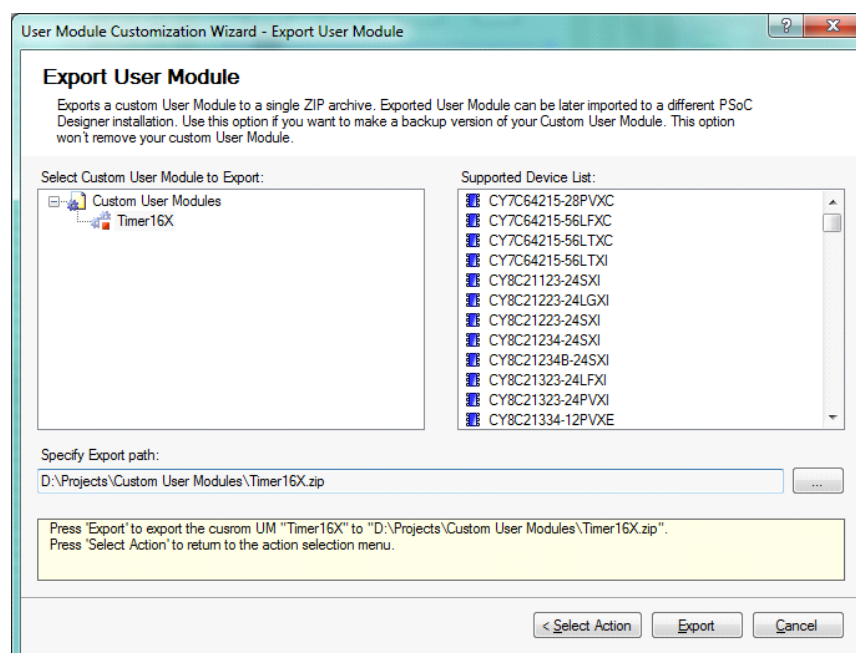
3. Select **Export User Module** option and select **Next**.



4. Select the **Timer16X** User Module.

5. Specify the path to which the user module will be exported.

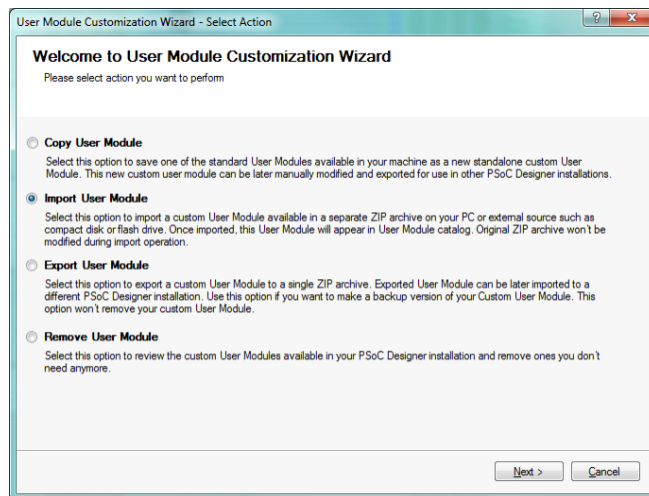
6. Select **Export**.



2.11.2 Importing the User Module

1. After the custom user module is exported, you can manually copy the Timer16X.zip archive to a specific location in another machine.
2. Launch PSoC Designer.
3. Go to **Tools > User Module Customization Wizard**.

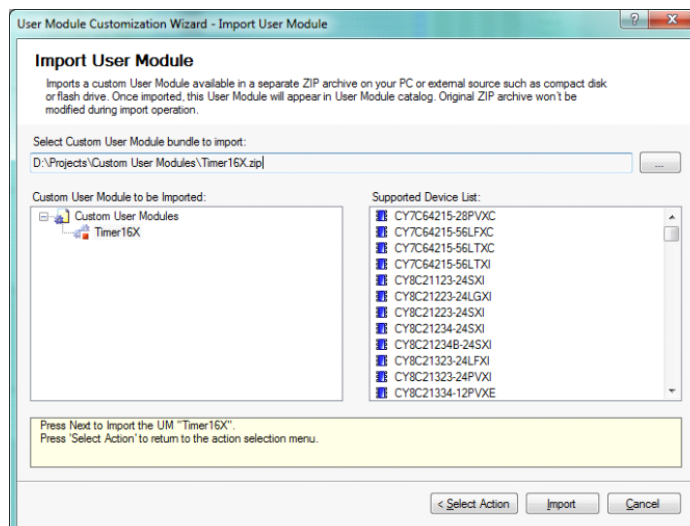
4. Select the **Import User Module** option.



5. Select **Next**.

6. Select the Timer16X.zip archive.

7. Select **Import**.

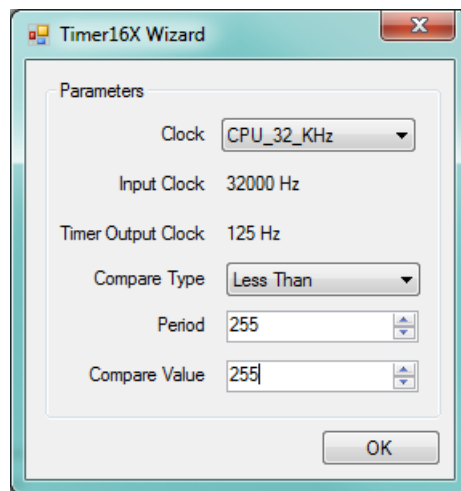


3. Extending the Custom User Module



The objective of this project is to build an improved Timer16 User Module, using the existing user module. The upper digital block of the present Timer16 has a parameter selection to generate interrupt on Capture, Terminal Count, and Compare True. The goal of the extending is to add to our Timer16X UM a Graphical User Interface for configuring the parameters (Wizard) as well as pre-processing, code generation placement, un-placement and Design Rule Checker scripts as described below.

- Wizard which allows configuring UM parameters with the next view:



'Clock', 'Period' and 'Compare Value', 'Compare Type' are UM parameters we want to configure. Range of values for 'Clock', 'Compare Type' parameters as well as minimum and maximum for 'Period' and 'Compare Value' to be loaded from UM architecture definition xml file. We will also display 'Input Clock' and 'Timer Output Clock' as read-only values.

Timer16X Wizard determines and saves time of last Wizard modifying. This value is used in *.h file as a constant. 'OK' Wizard button stores the parameters in our project.

- The pre-processing javascript (JS) allows us to form value list of 'CompareOut' UM parameter depending on the value selected for 'TerminalCountOut' parameter and filter this way row connections that are already being used.
- Placement JS that works during UM placement sets value of 'TC_PulseWidth' parameter to 'One-Half Clock' if doubling the SysClk is enabled ('SysClk*2 Disable' global parameter is set to 'No').
- Code generation JS which changes value of 'CompareValue' parameter if it exceeds 'Period' parameter value.
- Design Rule Check (DRC) JS displays a warning message in 'Output' window if calculated value of 'Timer Output Clock' is greater than 8 MHz. 'Timer Output Clock' parameter value depends on values of global parameters.

- Un-placement JS displays a message in separate dialog window to inform user that UM was deleted.

3.1 Add Wizard

Now we will add a Wizard to our UM. The Wizard configures '**Clock**', '**Period**' and '**Compare Value**', '**Compare Type**' parameters. Minimum and maximum ranges of both '**Period**' and '**Compare Value**' parameters are loaded from UM architecture definition xml file. The Wizard has 'Input Clock' and 'Timer Output Clock' read-only parameters which are calculated corresponding to the next:

- 'Input Clock' determine timer clock depends on 'Clock' UM parameters and values of 'Power Setting [Vcc / SysClk freq]', 'VC1= SysClk/N', 'VC2= VC1/N', 'VC3 Source' and 'VC3 Divider' correspondingly, in Hz;
- 'Timer Output Clock' = 'Input Clock'/'(Period'+1), in Hz

Also Wizard saves last time of its modifying. This value will be available in *.h file as a constant.

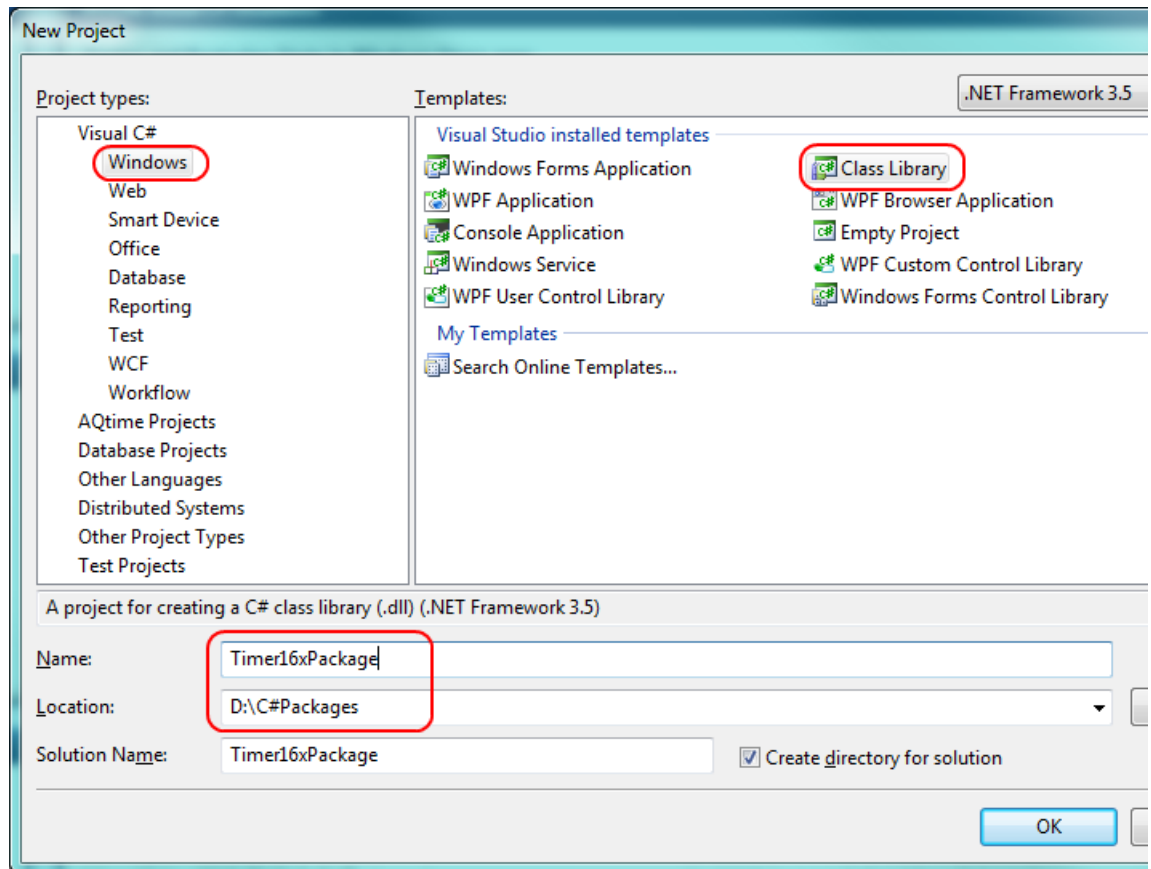
First, we will build a C# library that implements the Wizard GUI and behavior and then we will attach the Wizard to the UM, update UM xml description and *.h to manage constant.

3.1.1 Step 10: Create C# Package - General Rules

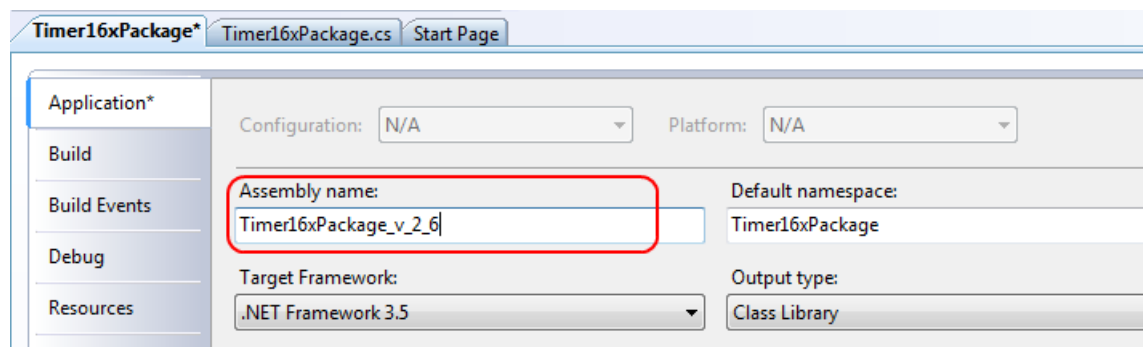
We will use Microsoft Visual Studio 2008 Service Pack 1 for Wizard creation. PSoC Designer is a .NET 2.0 applications so it is recommended to set 2.0 as the target framework for our library.

1. Launch Microsoft Visual Studio (VS).
2. Select **New > Project** option in **File** menu.
3. Select '**Windows**' project type and '**Class Library**' template in '**New Project**' dialog.
4. Set '**Timer16xPackage**' name in '**New Project**' dialog.

5. Set a location in '**New Project**' dialog to store created C# package.

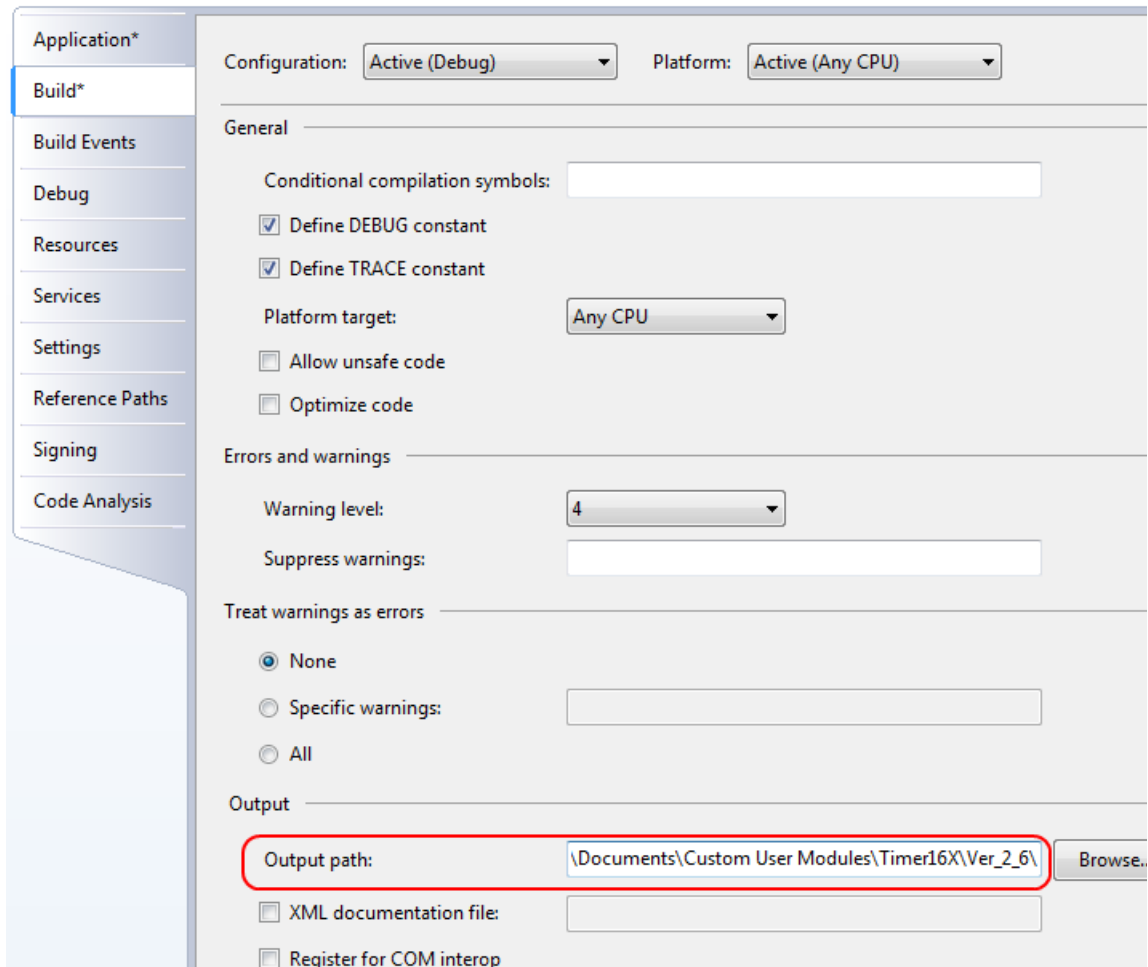


6. Rename '**Class1.cs**' in Solution Explorer to '**Timer16xPackage.cs**' (if Solution Explorer is not available by default, go to **View** menu and select '**Solution Explorer**' item).
7. Right-click on **Timer16xPackage** in Solution Explorer and select **Properties** in available menu.
8. Set '**Timer16xPackage_v_2_6**' assembly name in '**Application**' tab:



9. Set path to **Ver_2_6** folder of **Timer16X** custom UM in '**Build**' tab to point to the folder with custom user module folder as described below. In our case the path to the Timer16X Custom UM is "Doc-

uments\ Custom User Modules\”. Whenever you build your Wizard C# application, the DLL library file will appear updated in UM folder:



10. Save All.

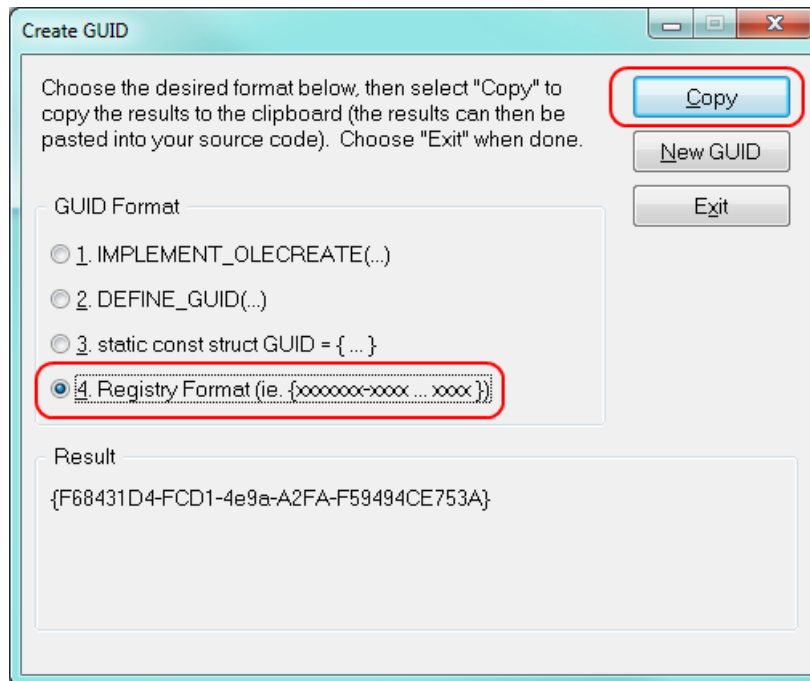
11. Open *Timer16xPackage.cs* file in VS.

12. Set the next line after namespace Timer16xPackage:

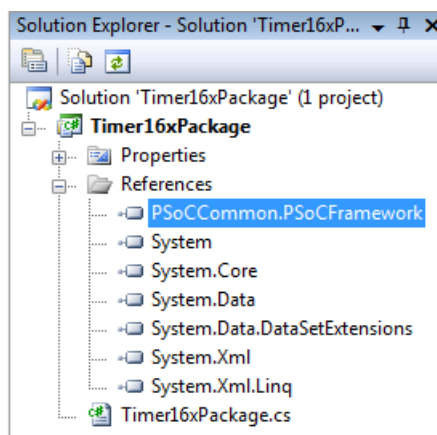
```
[Guid("XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"),
    ClassInterface(ClassInterfaceType.None)]
```

13. Select **Tools > Create GUID**.

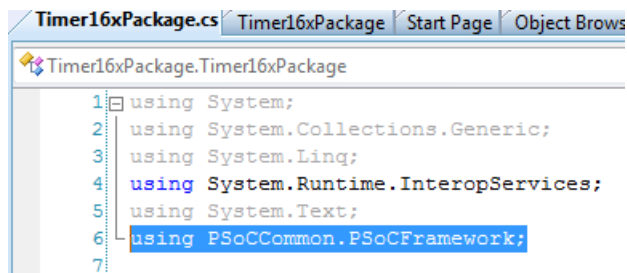
14. Select '**Registry Format**' in '**Create GUID**' dialog and select **Copy** button:



15. Exit from '**Create GUID**' dialog.
16. Replace `XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX` with `{ F68431D4 - FCD1 - 4e9a - A2FA - F59494CE753A }` in **Guid** string.
17. Remove '{' and '}' brackets in copied string.
18. Right-click on '**References**' in Solution Explorer and select '**Add Reference...**' item.
19. Go to '**Browse**' tab in '**Add Reference**' dialog.
20. Select '**PSoC Designer 5**' folder which is located in **PSoC Designer 5.4** installation folder.
21. Select '**PSoCCommon.PSoCFramework.dll**' file and close '**Add Reference**' dialog by pressing **OK**.
22. Check if **PSoCCommon.PSoCFramework** item is available under '**References**' in Solution Explorer:



23. Now open *Timer16xPackage.cs* file and add **PSoCCommon.PsoCFramework** name space to *Timer16xPackage.cs* file:

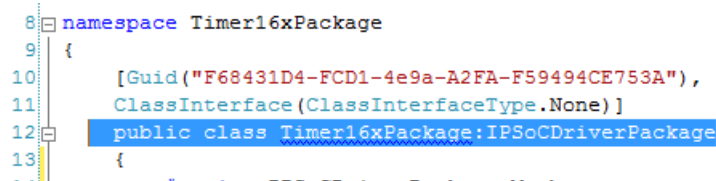


```

1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Runtime.InteropServices;
5 using System.Text;
6 using PSoCCommon.PSoCFramework;
7

```

24. Declare **IPSoCDriverPackage** interface in **Timer16xPackage** class definition:

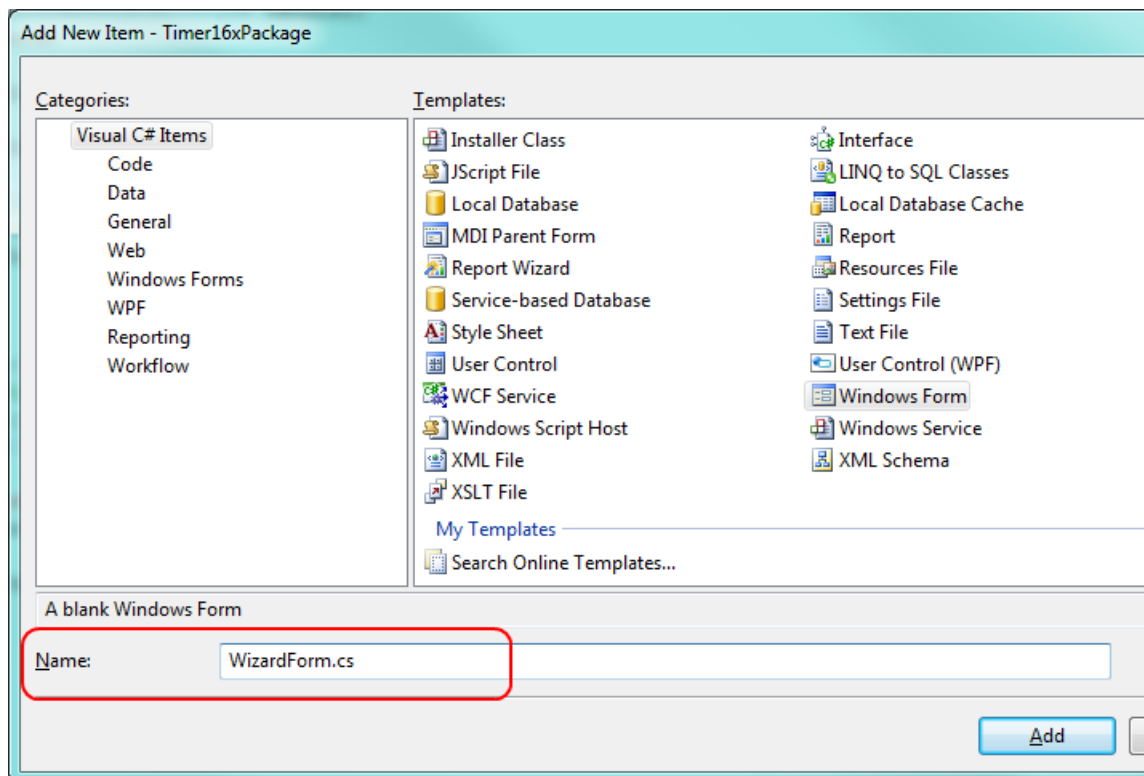


```

8 namespace Timer16xPackage
9 {
10     [Guid("F68431D4-FCD1-4e9a-A2FA-F59494CE753A"),
11     ClassInterface(ClassInterfaceType.None)]
12     public class Timer16xPackage:IPSoCDriverPackage
13     {
14         ...
15     }
16 }

```

25. Right-click on **IPSoCDriverPackage** interface and select **Interface > Implement Interface**.
26. Declare **_instanceName**, **_packageType**, **_cmxID**, **_lock** and **_installPath** private fields in **Timer16xPackage** class. See example of declaration in [Appendix A on page 55](#).
27. Right-click on **Timer16xPackage** in Solution Explorer and select **Add > Windows Form...**
28. Type '**WizardForm**' name in '**Add New Item - Timer16xPackage**' dialog.
29. Press '**Add**' button in '**Add New Item - Timer16xPackage**' dialog:



30. Implement **DoWizard** method in **Timer16xPackage** class. See example of declaration in [Appendix A on page 55](#).
31. Temporarily replace `WizardForm wizardForm = new WizardForm(_cmxID, _instanceName, sOverlayName);` string with: `WizardForm wizardForm = new WizardForm();` in **Timer16xPackage** class.
32. Implement **InstallPath**, **SetCmxID**, **SetInstanceName**, **SetPackageType** and **Start** methods in **Timer16xPackage** class. See example of declaration in [Appendix A on page 55](#).
33. Remove `throw new NotImplementedException();` string in **OnMessage** and **OnQuit** methods.
34. Implement **OnPropertiesUpdated** method in **Timer16xPackage** class. See example of declaration in [Appendix A on page 55](#).
35. Select **Build > Build Solution**. Check and correct any errors/warnings.

3.1.2 Step 11: Attach *.dll to UM

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27** UM sub-folder and open the *Tmr1627.xml* file
3. Replace **<XGUI_LIST>** element with the following text:

```
<XGUI_LIST>
<XGUI NAME="TIMER16X_WIZARD" DISPLAY_NAME="TIMER16X Wizard..."
SRC="..\Timer16xPackage_v_2_6.dll"/>
</XGUI_LIST>
```

Value of NAME attribute is free. The value of DISPLAY_NAME specifies text to be displayed when you right-click on UM in chip view. The value of SRC attribute specifies relative path to the library that implements the wizard.

Finally, the UM xml will look like this:

```

115 <PARAMETER NAME = "Period" TYPE = "BLOCK"
116 ORDER = "4" VALUE_TYPE = "INT"
117 MIN = "0x0000"
118 MAX = "0xffff"
119 HIDDEN = "TRUE"
120 >
135 <PARAMETER NAME = "CompareValue" TYPE = "BLOCK"
136 ORDER = "5" VALUE_TYPE = "INT"
137 MIN = "0x0000"
138 MAX = "0xffff"
139 HIDDEN = "TRUE"
140 >
155 <PARAMETER NAME = "CompareType"
156 ORDER = "6"
157 TYPE = "BLOCK"
158 HIDDEN = "TRUE"
159 >
191 <PARAMETER NAME = "ClockSync"
192 ORDER = "7"
193 TYPE = "BLOCK"
194 >
254 <PARAMETER NAME = "TC_PulseWidth" TYPE = "BLOCK"
255 ORDER = "8" SOURCE = "TIMER16_MSB"
256 REGISTER_NAME = "CONTROL_0"
257 BITFIELD = "TCPulseWide"
258 VALUE = "Full Clock"
259 />
260 </PARAMETER_LIST>
261
262 <API_REGISTER_ALIAS_LIST>
320
321 <XGUI_LIST>
322 <XGUI NAME="TIMER16X_WIZARD" DISPLAY_NAME="TIMER16X Wizard..." SRC="..\Time
323 </XGUI_LIST>
324
325 <API_FILE_LIST>
326 <API_FILE NAME = "Timer16INT.asm"
327 API_PREFIX = "Timer16"
328 API_PATH = "\"
329 />

```

3.2 Verify Wizard Availability in Custom UM

1. Launch PSoC Designer.
2. Create chip-level project based on CY8C29666-24LFXI part number (or on any other part from the Supported Devices List of our Timer16X UM) with PSoC Designer.
3. Place **Timer16X** UM.
4. Check if '**TIMER16X Wizard...**' appears in right-click menu of placed UM. If it does, press it. If there is no '**TIMER16X Wizard...**' present in right-click menu – close PD, go back and check the previous steps.
5. Observe empty wizard form appears upon press '**TIMER16X Wizard...**'

Now you are ready to implement the Wizard functionality. Whenever you add any code to the Wizard C# library, you can build and check to see if it works as expected. Remember to close PSoC Designer instances, so that the library can be updated properly in our UM folder.

3.2.1 Step 12: Update both xml and h files to manage time constant

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27** UM sub-folder and open the *Tmr1627.xml* file
3. Add the following after '**TC_PulseWidth**' parameter description:

```
<PARAMETER NAME="TimeLastModified" HIDDEN="TRUE" TYPE="API" ORDER="10"
SOURCE="NONE" VALUE_TYPE="STRING" VALUE="" />
```

4. CRET.

```

tr1627.xml | Timer16xRules.js | Tmr1627.h

<PARAMETER NAME = "CompareValue" TYPE = "BLOCK"
ORDER = "5" VALUE_TYPE = "INT"
MIN = "0x0000"
MAX = "0xffff"

<PARAMETER NAME = "CompareType"
ORDER = "6"
TYPE = "BLOCK"

<PARAMETER NAME = "ClockSync"
ORDER = "7"
TYPE = "BLOCK"

<PARAMETER NAME = "TC_PulseWidth" TYPE = "BLOCK"
ORDER = "8" SOURCE = "TIMER16_MSB"
REGISTER_NAME = "CONTROL_0"
BITFIELD = "TCPulseWide"
VALUE = "Full Clock"

<PARAMETER NAME="TimeLastModified" HIDDEN="TRUE" TYPE="API" ORDER="10" SOURCE="NONE"
</PARAMETER_LIST>

<API_REGISTER_ALIAS_LIST>
<API_REGISTER_ALIAS NAME = "COUNTER_LSB_REG"
SOURCE = "TIMER16_LSB"
REGISTER_NAME = "DATA_0"

<API_REGISTER_ALIAS NAME = "COUNTER_MSB_REG"
SOURCE = "TIMER16_MSB"
REGISTER_NAME = "DATA_0"

```

5. Open *Tmr1627.h* file in **CY8C27** UM sub-folder.

6. Add the next constant declaration after **COMPARE_VALUE**:

```

#define `@INSTANCE_NAME`_TIME_LAST_MODIFIED `@CTAB48` "`@TimeLastModified`"
//Last time Wizard was modified at this time

```

```

55 //
56 extern WORD w`@INSTANCE_NAME`_ReadCompareValue(void); // Deprecated
57 extern WORD w`@INSTANCE_NAME`_ReadTimerSaveCV(void); // Deprecated
58 extern WORD w`@INSTANCE_NAME`_ReadCounter(void); // Obsolete
59 extern WORD w`@INSTANCE_NAME`_ReadTimer(void); // Deprecated
60 extern WORD w`@INSTANCE_NAME`_CaptureCounter(void); // Obsolete
61
62
63 //-----
64 // Constants for `@INSTANCE_NAME` API's.
65 //-----
66
67 #define `@INSTANCE_NAME`_CONTROL_REG_START_BIT `@CTAB48` ( 0x01 )
68 #define `@INSTANCE_NAME`_INT_REG_ADDR `@CTAB48` ( 0x`@TIMER16_MSB_ISR_ADDR`
69 #define `@INSTANCE_NAME`_INT_MASK `@CTAB48` ( 0x`@TIMER16_MSB_ISR_MASK`
70 #define `@INSTANCE_NAME`_CaptureINT_MASK `@CTAB48` ( 0x`@TIMER16_LSB_CaptureI
71
72
73 //-----
74 // Constants for `@INSTANCE_NAME` user defined values
75 //-----
76
77 #define `@INSTANCE_NAME`_PERIOD `@CTAB48` ( 0x`@Period` )
78 #define `@INSTANCE_NAME`_COMPARE_VALUE `@CTAB48` ( 0x`@CompareValue` )
79 #define `@INSTANCE_NAME`_TIME_LAST_MODIFIED `@CTAB48` " `@TimeLastModified`" //La
80
81 //-----
82 // Register Addresses for `@INSTANCE_NAME`
83 //-----
84
85 `@COUNTER_LSB_REG_IOH` `@CTAB60`//Count register LSB
86 `@COUNTER_LSB_REG_H`
87 `@COUNTER_MSB_REG_IOH` `@CTAB60`//Count register MSB
88 `@COUNTER_MSB_REG_H`
89 `@PERIOD_LSB_REG_IOH` `@CTAB60`//Period register LSB
90 `@PERIOD_LSB_REG_H`
91 `@PERIOD_MSB_REG_IOH` `@CTAB60`//Period register MSB
92 `@PERIOD_MSB_REG_H`
93 `@COMPARE_LSB_REG_IOH` `@CTAB60`//Compare register LSB
94 `@COMPARE_LSB_REG_H`
    
```

3.2.2 Step 13: Implement Required Functionality in Wizard

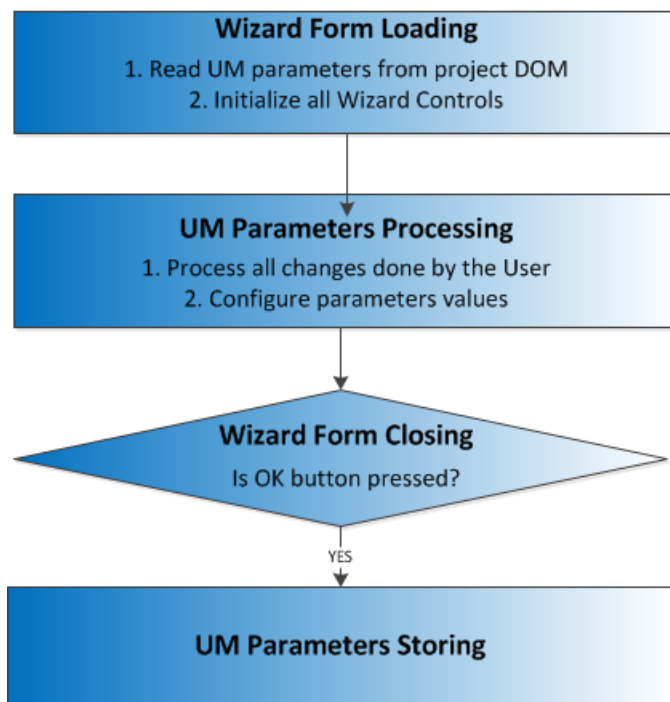
Now we have to write the C# code to implement the desired Wizard functionality. The complexity of that code depends on required features which should be implemented in Wizard. The target of this guide is to provide the example how to use the interfaces available for the Wizards in PSoC Designer. The example of code which implements Timer16X Wizard is represented in [Appendix A on page 55](#).

There are list of common rules and set of PSoC Designer commands which are used in UM Wizard implementation.

The following rules should be followed by Wizard author to achieve the UM Wizard requirements:

1. A current value, available value list, minimum and maximum range of UM parameters should be read from project DOM (Document Object Model) using WizardForm_Load . This must be done during Wizard form loading. See WizardForm_Load method in [Appendix A on page 55](#).
2. The parameters values should be stored in project DOM during Wizard form closing if OK button is pressed. See button_OK_Click method in [Appendix A on page 55](#). The 'Cancel' button handler should do nothing else than just close the form.
3. All controls on Wizard form should be initialized during Wizard form load. See code example of InitializeComboBox and InitializeNumericUpDown methods in [Appendix A on page 55](#).
4. There is no need to store individual parameter value immediately in Wizard code – you can save all parameters in the project DOM later, all at once;
5. Handle PropertiesUpdated event after Wizard form closing if **OK** button is pressed to reflect UM parameters values on interconnect view and in Parameter Window.

The following diagram represents Wizard functionality in general:



[Table 3-1 on page 39](#) represents commands which can be used in UM Wizard. All these commands are used as second argument of CMXEngineAccessor.GetData method. There are details of CMX-EngineAccessor.GetData method:

1. Namespace: PSoCCommon.PsoCFramework
2. Parameters:
 - a. Parameter #1: project ID;
 - b. Parameter #2: command;
 - c. Parameter#3,#4 and #5 depend on command;
3. Return object: depend on command, but always in string format.

The following example demonstrates how you can use CMXEngineAccessor.GetData method in UM Wizard:

```
string sOverlayName = CMXEngineAccessor.GetData(_cmxID,
"GET_OVERLAY_NAME_FROM_INSTANCE", _instanceName, "", "");
```

Table 3-1. CMXEngineAccessor.GetData Commands

#	Command	Parameters	Comments
1	GET_OVERLAY_NAME_FROM_INSTANCE	Parameter#3 - UM instance name Parameter#4 - empty string Parameter#5 - empty string Return - project name	Returns project name
2	GET_USERMODULE_PARAMETERS	Parameter#3 - project name Parameter#4 - UM instance name Parameter#5 - empty string Return - UM parameter DOM	Returns UM parameter DOM which contains current values, value list, maximum and minimum ranges
3	SET_USERMODULE_PARAM_NO_OVERLAY	Parameter#3 - UM instance name Parameter#4 - UM parameter name Parameter#5 - UM parameter value	Sets UM parameter value in project DOM
4	GET_GLOBAL_RESOURCES	Parameter#3 - project name Parameter#4 - UM instance name Parameter#5 - empty string Return - global parameters DOM	Returns global parameters DOM which contains current value of parameters
5	GET_DEVICE_DISPLAY_NAME	Parameter#3 - empty string Parameter#4 - empty string Parameter#5 - empty string Return - name of part number	Returns name of PSoC Device part number
6	GET_USERMMODULE_INFO_FROM_INSTANCENAME	Parameter#3 - UM instance name Parameter#4 - empty string Parameter#5 - empty string Return - Multi UM DOM	Returns DOM of Multi UM which includes path to custom UM, UM name, Multi UM name and etc..
7	GET_CURRENT_PIN_INFO	Parameter#3 - project name Parameter#4 - empty string Parameter#5 - empty string Return - pins DOM	Returns DOM of all pins which includes pin name, pin label and pin custom name of each of them.
8	GET_UM_ROOT_PATH	Parameter#3 - UM instance name Parameter#4 - empty string Parameter#5 - empty string Return - path to UM	Returns path to folder inside UM folder where UM package is located

Table 3-1. CMXEngineAccessor.GetData Commands (*continued*)

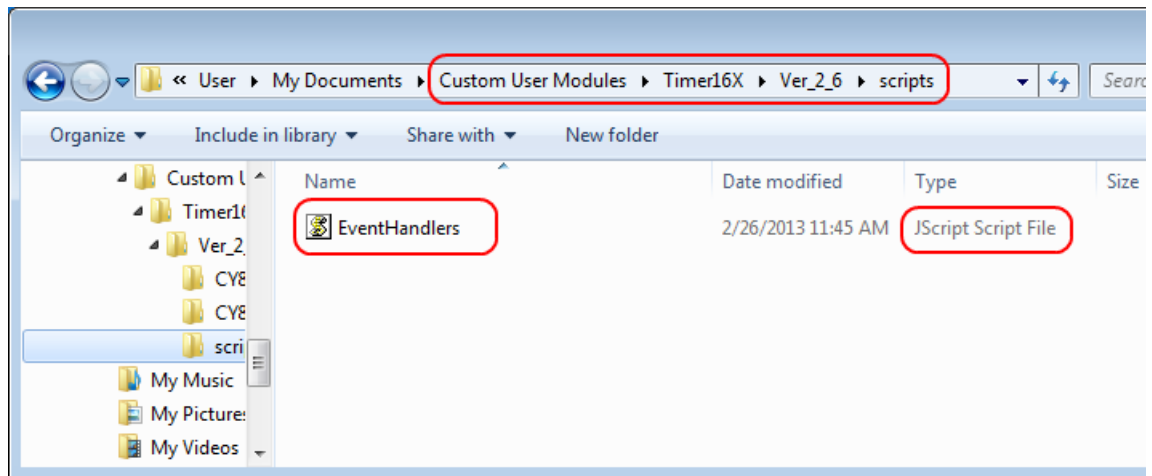
#	Command	Parameters	Comments
9	GET_UM_PATH	Parameter#3 - UM instance name Parameter#4 - empty string Parameter#5 - empty string Return - path to UM	Returns path to UM xml file which describes current UM
10	SET_RESERVED_RESOURCE	Parameter#3 - project name Parameter#4 - name of reserved resource Parameter#5 - value of reserved resource	Sets value of reserved resource (like AnalogMuxBus, Decimator)
11	GET_RESERVED_RESOURCE_CURRENT_VAL	Parameter#3 - project name Parameter#4 - name of reserved resource Parameter#5 - empty string Return - current value	Gets current value of reserved resource (like AnalogMuxBus, Decimator, Row_0_Output_1)
12	GET_PIN_INFO_VECTOR	Parameter#3 - project name Parameter#4 - empty string Parameter#5 - empty string Return - pins DOM	Returns DOM of all pins which includes type, interrupt and initial value list and current value.
13	CMX_GET_MEMORY_CAPACITY	Parameter#3 - project name Parameter#4 - empty string Parameter#5 - empty string Return - memory DOM	Returns DOM which includes RAM/ROM settings.
14	GET_DEVICE_NAME	Parameter#3 - project name Parameter#4 - UM instance name Parameter#5 - empty string Return - ID of part number	Returns internal ID of PSoC Device part number

3.3 Add Pre-Processing Javascript

An adding of pre-processing JS to UM requires a modifying of UM xml file and creating JS code with required functionality.

3.3.1 Step 14: Create JS file

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Create scripts folder on the **CY8C27** folder level.
3. Go to **scripts** folder.
4. Create file and name it *EventHandlers.js*.



3.3.2 Step 15: Attach Pre-Processing Javascript to UM

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27** UM sub-folder and open the *Tmr1627.xml* file
3. Go to '**CompareOut**' parameter description.
4. Add the pre-processing JS call:

```
<XGUI_PARAM>
  <PRE_PROCESS_LIST>
    <PRE_PROCESS_SRC="..\scripts\EventHandlers.js" CALL="CompareOutSelect"
ORDER="0"/>
  </PRE_PROCESS_LIST>
</XGUI_PARAM>
```

The value of **SRC** attribute specifies name of JS file which implements pre-processing and relative path to it. The value of **CALL** attribute determines name of function which is called before value of '**CompareOut**' parameter gets changed. The value of **ORDER** attribute specifies order of pre-processing calls for current parameter.

```

84         REGISTER_NAME = "DIG_Input"
85         BITFIELD = "ClockSelect"
86     >
99     <PARAMETER NAME = "Capture" TYPE = "INPUT"
100         ORDER = "1" SOURCE = "TIMER16_LSB"
101         REGISTER_NAME = "DIG_Input"
102         BITFIELD = "DataSelect"
103     />
104     <PARAMETER NAME = "TerminalCountOut" TYPE = "OUTPUT"
105         ORDER = "2" SOURCE = "TIMER16_MSB"
106         REGISTER_NAME = "DIG_Output"
107         BITFIELD = "DigOutputSelect"
108     />
109     <PARAMETER NAME = "CompareOut" TYPE = "OUTPUT"
110         ORDER = "3" SOURCE = "TIMER16_MSB"
111         REGISTER_NAME = "DIG_Output"
112         BITFIELD = "AUXOutputSelect"
113     />
114     <XGUI_PARAM>
115         <PRE_PROCESS_LIST>
116         <PRE_PROCESS SRC="..\scripts\EventHandlers.js" CALL="Co
117         </PRE_PROCESS_LIST>
118     </XGUI_PARAM>
119     </PARAMETER>
120     <PARAMETER NAME = "Period" TYPE = "BLOCK"
121         ORDER = "4" VALUE_TYPE = "INT"
122         MIN = "0x0000"
123         MAX = "0xffff"
124     >
139     <PARAMETER NAME = "CompareValue" TYPE = "BLOCK"
140         ORDER = "5" VALUE_TYPE = "INT"
141         MIN = "0x0000"
142         MAX = "0xffff"
143     >

```

Note: A post-processing JS to custom UM (see Step #14-15) can be added the same way. In this case you should add PRE_PROCESS_LIST element to UM parameter in Step #15.

3.3.3 Step 16: Implement Required Functionality in Pre-/Post-Processing JS

The pre-processing JS forms value list of 'CompareOut' UM parameter depending on the value selected for 'TerminalCountOut' parameter and filter this way row connections that are already being used.

The example of code which implements the filtering for 'CompareOut' parameter is shown in [Appendix B on page 66](#).

Pay attention that post- and pre-processing JS function has one argument which contains DOM of processed UM and its parameters. This DOM contains the following attributes:

- UM_INST_NAME - UM instance name;
- UM_NAME - UM name;
- PARAM_NAME - name of processed UM parameter;
- PARAM_VALUE - value of processed UM parameter.

There is a set of PSoC Designer methods and objects that can be used in UM pre-/post-processing JS. Table 3-2 represents the objects used in custom pre-/post-processing JS.

Table 3-2. Pre-/Post-Processing JS Global Objects

Object	Comments
PS	General Purpose Automation Interface. This object dispatches pointer to IPSoCottoHelper interface and supports the following methods:
	<code>object GetXMLDOM();</code>
	<code>string ReadTextFile(string path);</code>
	<code>string WriteTextFile(string path, string text);</code>
	<code>object GetDeviceEditor();</code>
	<code>object GetProjectManager();</code>
	<code>object GetDebugTarget();</code>
	<code>object GetBuildManager();</code>
	<code>short FileExists(string path);</code>
	<code>long CreateDirectory(string path);</code>
	<code>long DeleteFile(string path);</code>
	<code>string ReadFile(string path);</code>
	<code>long WriteFile(string path, string data);</code>
	<code>long GetLastError();</code>
	<code>string FormatMessage(long id);</code>
	<code>long CopyFile(string from, string to, short failIfExists);</code>
	<code>long GetMainWnd();</code>
	<code>long FindWindow(long parentWnd, long childWnd, string className, string windowName);</code>
	<code>long SendMessage(long wnd, long msg, string wParamType, string wParam, string lParamType, string lParam, long post);</code>
	<code>long SendMessageToAllDevEditViews(long msg, string wParamType, string wParam, string lParamType, string lParam, long post);</code>
	<code>long RegSetValueEx(string key, string subKey, string valueName, string type, string data);</code>
	<code>long RegQueryValueEx(string key, string subKey, string valueName, string type, ref string data);</code>
	<code>long ShowMessageBox(string msg);</code>
	<code>long StartProcess(string cmdLine);</code>
	<code>string GetShortPathFromLong(string path);</code>
	<code>long RegQueryValueEx(string key, string subKey, string valueName, string type, ref string data);</code>
otto	Device Editor Document Interface. This object dispatches pointer to IDeviceEditorDocument interface and supports the following methods:
	<code>string GetCurUMParamValue(string param);</code>
	<code>int SetCurUMParamValue(string param, string val);</code>
	<code>string GetCurUMParamValueRange(string param);</code>
	<code>int SetCurUMParamValueRange(string param, string valRange);</code>
	<code>string GetGlobalResourceValue(string param);</code>
	<code>int SetGlobalResourceValue(string param, string val);</code>
	<code>string GetGlobalResourceValueRange(string param);</code>
	<code>int SetGlobalResourceValueRange(string param, string valRange);</code>
	<code>string CallMethod(string type, string callBack, string paramList, string wantReturn);</code>

The most usable method of PS interface is GetDeviceEditor() which returns Device Editor Automation Interface which has set of useful methods:

```
var devEditor = PS.GetDeviceEditor();
```

Table 3-3. List of Device Editor Methods

Object	Comments
PS.GetDeviceEditor();	Device Editor Automation Interface. This object dispatches pointer to IPSoCDeviceEditorOtto interface and supports the following methods:
	<code>int GenerateSrc();</code>
	<code>int SelectUserModule(string szName, string szInstanceName);</code>
	<code>int UnSelectUserModule(string szName);</code>
	<code>int PlaceUserModule(string szName);</code>
	<code>int SetUserModuleParameter(string szInstanceName, string szValueName, string szValueValue);</code>
	<code>int NextShape(string szInstanceName);</code>
	<code>int ClearConfiguration();</code>
	<code>int ClearUserModuleParameter(string szInstanceName, string szParamName);</code>
	<code>int lGenerateSrc();</code>
	<code>int lSelectUserModule(string szName, string szInstanceName);</code>
	<code>int lUnSelectUserModule(string szName);</code>
	<code>int lPlaceUserModule(string szName);</code>
	<code>int lSetUserModuleParameter(string szInstanceName, string szValueName, string szValueValue);</code>
	<code>int lNextShape(string szInstanceName);</code>
	<code>int lClearConfiguration();</code>
	<code>int lClearUserModuleParameter(string szInstanceName, string szParamName);</code>
	<code>int SetUserModuleParameter2(string szOverLayName, string szInstanceName, string szValueName, string szValueValue);</code>
	<code>int lSetUserModuleParameter2(string szOverLayName, string szInstanceName, string szValueName, string szValueValue);</code>
	<code>string GetUserModuleParameter2(string szOverLayName, string szInstanceName, string szValueName);</code>
	<code>string GetUserModule(string szOverLayName, string szBlkName);</code>
	<code>string GetPinGrid(string szOverLayName);</code>
	<code>string GetUserModulePath(string szUMName);</code>
	<code>string GetDevicePath(string lpszDeviceName);</code>
	<code>string GetBaseDevice();</code>
	<code>int EnumerateConfigs(object lpdCallbackTarget, string lpszCallBack);</code>
	<code>int EnumerateSelectedUMs(object lpdCallbackTarget, string lpszConfigName, string lpszCallBack);</code>
	<code>string GetDBMainPath();</code>
	<code>string GetGlobalResourceValue(string lpszOverLayName, string lpszResourceName);</code>
	<code>int EnumerateUMParams(object lpdCallbackTarget, string lpszConfigName, string strUMInstName, string strUMName, string lpszCallBack);</code>
	<code>short IsUMPlaced(string lpszConfigName, string lpszUMInstName);</code>
	<code>int lSetGlobalResourceValue(string sz1, string sz2, string sz3);</code>

Table 3-3. List of Device Editor Methods (*continued*)

Object	Comments
	<code>string GetReservedResourceValue(string sz1, string sz2);</code>
	<code>int lSetReservedResourceValue(string sz1, string sz2, string sz3);</code>
	<code>int lSelectUserModule2(string sz1, string sz2, string sz3);</code>
	<code>int lUnSelectUserModule2(string sz1, string sz2);</code>
	<code>int lPlaceUserModule2(string sz1, string sz2);</code>
	<code>int lUnPlaceUserModule2(string sz1, string sz2);</code>
	<code>int lAddConfigOverlay(string sz1);</code>
	<code>int lDeleteConfigOverlay(string sz1);</code>
	<code>int SetMultipleUserModule(string sOverlayName, string sInstanceName, string sMultiModuleName);</code>
	<code>string GetDeviceName();</code>
	<code>short IsUMPlaceable(string lpszConfigName, string lpszUMInstName);</code>
	<code>int EnumUnsyncDigitalBlocks(object lpdCallbackTarget, string lpszConfigName, string strUMInstName, string strUMName, string lpszCallBack);</code>
	<code>int MoveUserModuleShapeToBlock(string sz1, string sz2, string sz3);</code>
	<code>int MoveUserModuleShapeToBlock2(string szOverlayName, string szInstanceName, string szBlockName, string szShapeBlkName);</code>
	<code>int NextShape2(string szOverlayName, string szInstanceName);</code>
	<code>int lNextShape2(string szOverlayName, string szInstanceName);</code>
	<code>string GetConfigInfo(string szPath);</code>
	<code>int OpenConfiguration(string szPath);</code>
	<code>string GetDeviceDisplayName();</code>
	<code>string GetPartIncludeFolder(string szDevName);</code>
	<code>string GetPartTemplateFolder(string szDevName);</code>
	<code>string GetMemoryCapacity();</code>
	<code>int EnumerateConfigsXGUI(object lpdCallbackTarget, string lpszCallBack);</code>
	<code>int EnumerateSelectedUMsXGUI(object lpdCallbackTarget, string lpszConfigName, string lpszCallBack);</code>
	<code>string GetData(string sCmd, string sArg1, string sArg2, string srArg3, string sArg4);</code>
	<code>string EnumerateConfigs2();</code>
	<code>string EnumerateSelectedUMs2(string lpszConfigName);</code>
	<code>string EnumerateUMParams2(string lpszConfigName, string lpszUMInstName, string lpszUMName);</code>
	<code>string GetUserModuleFilePathName(string szUMName);</code>
	<code>int RefreshGlobalResources(string configName);</code>
	<code>int RefreshReservedResources(string configName);</code>
	<code>string GetVersionedUserModulePath(string szUMName, string umVersion);</code>
	<code>string GetVersionedUserModulePathName(string szUMName, string umVersion);</code>

You may use the commands from [Table 3-1 on page 39](#) for Device Editor `GetData ()` method, but pay attention that in this case command is the first argument:

```
var devEditor = PS.GetDeviceEditor();
```

```
var sOverlay = devEditor.GetData("GET_OVERLAY_NAME_FROM_INSTANCE", sInstName,
    "", "", "");
```

3.4 Add Placement Javascript

Now we will add a placement JS to our UM. Placement JS sets value of '**TC_PulseWidth**' parameter to '**One-Half Clock**' if doubling the SysClk is enabled ('**SysClk*2 Disable**' global parameter is set to '**No**').

3.4.1 Step 17: Attach Placement Javascript to UM

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27** UM sub-folder and open the *Tmr1627.xml* file
3. Go to **XGUI_LIST** element description.
4. Add the following after **TIMER16X_WIZARD** XGUI element:

```
<XGUI NAME = "PlaceScript" SRC = "..\scripts\EventHandlers.js" TYPE="UM_PLACE"
>
    <XGUI_FUNC_LIST>
        <XGUI_FUNC FUNC_NAME="Timer16xPlaceUM" />
    </XGUI_FUNC_LIST>
</XGUI>
```

The value of **NAME** attribute is free. A value of **SRC** attribute specifies name of JS file with a function that implements changing value list of 'Clock' UM parameter during UM placing and relative path to it. The value of **FUNC_NAME** attribute determines name of function which is called. The value of **TYPE** attribute should be **UM_PLACE**.

```

297 <API_REGISTER_ALIAS NAME = "FUNC_LSB_REG"
298     SOURCE = "TIMER16_LSB"
299     REGISTER_NAME = "DIG_BasicFunction"
300 />
301 <API_REGISTER_ALIAS NAME = "FUNC_MSB_REG"
302     SOURCE = "TIMER16_MSB"
303     REGISTER_NAME = "DIG_BasicFunction"
304 />
305 <API_REGISTER_ALIAS NAME = "INPUT_LSB_REG"
306     SOURCE = "TIMER16_LSB"
307     REGISTER_NAME = "DIG_Input"
308 />
309 <API_REGISTER_ALIAS NAME = "INPUT_MSB_REG"
310     SOURCE = "TIMER16_MSB"
311     REGISTER_NAME = "DIG_Input"
312 />
313 <API_REGISTER_ALIAS NAME = "OUTPUT_LSB_REG"
314     SOURCE = "TIMER16_LSB"
315     REGISTER_NAME = "DIG_Output"
316 />
317 <API_REGISTER_ALIAS NAME = "OUTPUT_MSB_REG"
318     SOURCE = "TIMER16_MSB"
319     REGISTER_NAME = "DIG_Output"
320 />
321 </API_REGISTER_ALIAS_LIST>
322
323 <XGUI_LIST>
324     <XGUI NAME="TIMER16X WIZARD" DISPLAY_NAME="TIMER16X Wizard..." SRC="
325     <XGUI NAME = "PlaceScript" SRC = "..\scripts\EventHandlers.js" TYPE=
326     <XGUI_FUNC_LIST>
327         <XGUI_FUNC FUNC_NAME="Timer16xPlaceUM" />
328     </XGUI_FUNC_LIST>
329 </XGUI>
330 </XGUI_LIST>
331
332 <API_FILE_LIST>
333     <API_FILE NAME = "Timer16INT.asm"
334     API_PREFIX = "Timer16"
335     API_PATH = "\"
336 />

```

3.4.2 Step #18: Implement Required Functionality in Placement JS

1. Go to **scripts** folder and open *EventHandlers.js* file.
2. Add **Timer16xPlaceUM** function implementation.

Pay attention that placing function has one argument which contains instance name of processed UM.

The example of code which initializes the '**TC_PulseWidth**' parameter is represented in [Appendix B on page 66](#).

Only **PS** global object which is listed in [Table 3-2 on page 43](#) is available in code generation JS. All methods from [Table 3-2 on page 43](#) and [Table 3-3 on page 44](#) may be used in this type of JS.

3.5 Add Code Generation Javascript

An adding of code generation JS to UM require a modifying of UM xml file and creating JS code with required functionality. Code generation JS changes value of 'CompareValue' parameter if it exceeds 'Period' parameter value.

3.5.1 Step 19: Attach Code Generation Javascript to UM

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27** UM sub-folder and open the *Tmr1627.xml* file
3. Go to **XGUI_LIST** element description.
4. Add the following after PlaceScript element:

```
<XGUI NAME="CodeGenScript" SRC="..\scripts\EventHandlers.js"
TYPE="UM_CODE_GEN">
  <XGUI_FUNC_LIST>
    <XGUI_FUNC FUNC_NAME="Timer16xGenerateCode"/>
  </XGUI_FUNC_LIST>
</XGUI>
```

The value of **NAME** attribute is free. A value of **SRC** attribute specifies name of JS file where function which implements changing value of 'CompareValue' UM parameter, and relative path to it. The value of **FUNC_NAME** attribute determines name of function which is called. The value of **TYPE** attribute should be **UM_CODE_GEN**.


```

289 <API_REGISTER_ALIAS NAME = "CONTROL_LSB_REG"
290     SOURCE = "TIMER16_LSB"
291     REGISTER_NAME = "CONTROL_0"
292 />
293 <API_REGISTER_ALIAS NAME = "CONTROL_MSB_REG"
294     SOURCE = "TIMER16_MSB"
295     REGISTER_NAME = "CONTROL_0"
296 />
297 <API_REGISTER_ALIAS NAME = "FUNC_LSB_REG"
298     SOURCE = "TIMER16_LSB"
299     REGISTER_NAME = "DIG_BasicFunction"
300 />
301 <API_REGISTER_ALIAS NAME = "FUNC_MSB_REG"
302     SOURCE = "TIMER16_MSB"
303     REGISTER_NAME = "DIG_BasicFunction"
304 />
305 <API_REGISTER_ALIAS NAME = "INPUT_LSB_REG"
306     SOURCE = "TIMER16_LSB"
307     REGISTER_NAME = "DIG_Input"
308 />
309 <API_REGISTER_ALIAS NAME = "INPUT_MSB_REG"
310     SOURCE = "TIMER16_MSB"
311     REGISTER_NAME = "DIG_Input"
312 />
313 <API_REGISTER_ALIAS NAME = "OUTPUT_LSB_REG"
314     SOURCE = "TIMER16_LSB"
315     REGISTER_NAME = "DIG_Output"
316 />
317 <API_REGISTER_ALIAS NAME = "OUTPUT_MSB_REG"
318     SOURCE = "TIMER16_MSB"
319     REGISTER_NAME = "DIG_Output"
320 />
321 </API_REGISTER_ALIAS_LIST>
322
323 <XGUI_LIST>
324     <XGUI NAME="TIMER16X_WIZARD" DISPLAY_NAME="TIMER16X Wizard..." SRC="..\Timer16xPackage_
325     <XGUI NAME = "PlaceScript" SRC = "..\scripts\EventHandlers.js" TYPE="UM_PLACE" >
326     <XGUI NAME="CodeGenScript" SRC="..\scripts\EventHandlers.js" TYPE="UM_CODE_GEN">
327     <XGUI_FUNC_LIST>
328     <XGUI_FUNC FUNC_NAME="Timer16xGenerateCode"/>
329     </XGUI_FUNC_LIST>
330     </XGUI>
331 </XGUI_LIST>
332
333 <API_FILE_LIST>
334     <API_FILE NAME = "Timer16INT.asm"
335     API_PREFIX = "Timer16"
336     API_PATH = "\"
337 />
338 <API_FILE NAME = "Tmr1627.inc" />
339 <API_FILE NAME = "Tmr1627.asm" />
340 <API_FILE NAME = "Tmr1627.h" />
341 </API_FILE_LIST>
342 </USER_MODULE>
343 </USER_MODULE_LIST>
344 </PSOC_DEVICE_DB>
    
```

5. CRET.

3.5.2 Step 20: Implement Required Functionality in Code Generation JS

1. Go to **scripts** folder and open *EventHandlers.js* file.
2. Add **Timer16xGenerateCode** function implementation.

Ensure that code generation function has one argument which contains instance name of processed UM. The example of code which changes value of '**CompareValue**' parameter is represented in [Appendix B on page 66](#).

Only **PS** global object which is listed in [Table 3-2 on page 43](#) is available in code generation JS. All methods from [Table 3-2 on page 43](#) and [Table 3-3 on page 44](#) may be used in this type of JS.

3.6 Add Design Rule Check Javascript

An adding of DRC JS to UM requires modifying a UM xml file and creating JS code with required functionality. DRC displays a warning message in 'Output' window if calculated value of '**Timer Output Clock**' is greater than 8 MHz. '**Timer Output Clock**' parameter value depends on values of global parameters. See [Add Wizard on page 28](#) for details. Text message is: '*Timer Output Clock for UM_NAME is greater than 8 MHz. That is why UM_NAME can not be connected to internal source.*' where **UM_NAME** will be replaced with current UM instance name.

Follow [Step 14: Create JS file on page 41](#) to create JS file for DRC implementation. Name this file *Timer16xRules.js*. We recommend using separate file for DRC, but it is not obligatory. You may use the same JS file (*EventHandlers.js*) where placement, un-placement, code generation and pre-/post-processing scripts are implemented.

3.6.1 Step 21: Attach Design Rule Check Javascript to UM

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27Timer** UM sub-folder and open the *Tmr1627.xml* file
3. Add the following after **API_FILE_LIST** element:

```
<RULE_LIST>
  <RULE CODE="" PATH="..\scripts\Timer16xRules.js" CALL="CheckTimerOutput-
Clock" SEVERITY="1"/>
</RULE_LIST>
```

The value of **CODE** attribute is free and may be empty string. A value of **PATH** attribute specifies name of JS file which contains DRC function implementation, and relative path to it. The value of **CALL** attribute determines name of function which is called. The value of **SEVERITY** attribute defines severity of the Rule and possible values are numeric from 1 to 5.

Note: If original UM has DRC, the actual path is set in **PATH** attribute. In this case you should update actual path to relative in **PATH** attribute like in example above.

```

303 <API_REGISTER_ALIAS NAME = "FUNC_LSB_REG"
304     SOURCE = "TIMER16_LSB"
305     REGISTER_NAME = "DIG_BasicFunction"
306 />
307 <API_REGISTER_ALIAS NAME = "FUNC_MSB_REG"
308     SOURCE = "TIMER16_MSB"
309     REGISTER_NAME = "DIG_BasicFunction"
310 />
311 <API_REGISTER_ALIAS NAME = "INPUT_LSB_REG"
312     SOURCE = "TIMER16_LSB"
313     REGISTER_NAME = "DIG_Input"
314 />
315 <API_REGISTER_ALIAS NAME = "INPUT_MSB_REG"
316     SOURCE = "TIMER16_MSB"
317     REGISTER_NAME = "DIG_Input"
318 />
319 <API_REGISTER_ALIAS NAME = "OUTPUT_LSB_REG"
320     SOURCE = "TIMER16_LSB"
321     REGISTER_NAME = "DIG_Output"
322 />
323 <API_REGISTER_ALIAS NAME = "OUTPUT_MSB_REG"
324     SOURCE = "TIMER16_MSB"
325     REGISTER_NAME = "DIG_Output"
326 />
327 </API_REGISTER_ALIAS_LIST>
328
329 <XGUI_LIST>
330     <XGUI NAME="TIMER16X_WIZARD" DISPLAY_NAME="TIMER16X Wizard..." SRC="" />
331     <XGUI NAME = "PlaceScript" SRC = "..\scripts\EventHandlers.js" />
332     <XGUI NAME = "CodeGenScript" SRC="..\scripts\EventHandlers.js" />
333     <XGUI NAME = "UnPlaceScript" SRC = "..\scripts\EventHandlers.js" />
334 </XGUI_LIST>
335
336 <API_FILE_LIST>
337     <API_FILE NAME = "Timer16INT.asm"
338     API_PREFIX = "Timer16"
339     API_PATH = "\"
340 />
341     <API_FILE NAME = "Tmr1627.inc" />
342     <API_FILE NAME = "Tmr1627.asm" />
343     <API_FILE NAME = "Tmr1627.h" />
344 </API_FILE_LIST>
345 <RULE_LIST>
346     <RULE CODE="" PATH="..\scripts\Timer16xRules.js" CALL="CheckTimer" />
347 </RULE_LIST>
348 </USER_MODULE>
349 </USER_MODULE_LIST>
350 </PSOC_DEVICE_DB>

```

4. CRET.

3.6.2 Step 22: Implement Required Functionality in Design Rule Check JS

1. Go to **scripts** folder and open *Timer16xRules.js* file.
2. Add **CheckTimerOutputClock** function implementation.

Be sure that the DRC function has one argument which contains instance name of processed UM. The example of code which display message in 'Output' window during project generation is represented in [Appendix C on page 68](#).

Only **PS** global object which is listed in [Table 3-2 on page 43](#) is available DRC JS. All methods from [Table 3-2 on page 43](#) and [Table 3-3 on page 44](#) may be used in this type of JS. There is one limitation - you cannot change current values, filter value list of UM parameters, change current values of global or reserved resources. You can only read current values and display a warning message.

3.7 Add Un-placement Javascript

An adding of un-placement JS to UM require a modifying of UM xml file and creating JS code with required functionality. Un-placement JS displays information message to notice user about UM removing.

3.7.1 Step 23: Attach Un-placement Javascript to UM

1. Go to the **Timer16X** folder in the folder with custom UMs.
2. Open the **CY8C27** UM sub-folder and open the *Tmr1627.xml* file
3. Go to **XGUI_LIST** element description.
4. Add the following after CodeGenScript element:

```
<XGUI NAME = "UnPlaceScript" SRC = "..\scripts\EventHandlers.js"
  TYPE="UM_UNPLACE" >
  <XGUI_FUNC_LIST>
    <XGUI_FUNC FUNC_NAME="Timer16xUnPlaceUM" />
  </XGUI_FUNC_LIST>
</XGUI>
```

The value of **NAME** attribute is free. A value of **SRC** attribute specifies name of JS file where function which displays information message, and relative path to it. The value of **FUNC_NAME** attribute determines name of function which is called. The value of **TYPE** attribute should be **UM_UNPLACE**.

```

305 REGISTER_NAME = "DIG_BasicFunction"
306 />
307 <API_REGISTER_ALIAS NAME = "FUNC_MSB_REG"
308 SOURCE = "TIMER16_MSB"
309 REGISTER_NAME = "DIG_BasicFunction"
310 />
311 <API_REGISTER_ALIAS NAME = "INPUT_LSB_REG"
312 SOURCE = "TIMER16_LSB"
313 REGISTER_NAME = "DIG_Input"
314 />
315 <API_REGISTER_ALIAS NAME = "INPUT_MSB_REG"
316 SOURCE = "TIMER16_MSB"
317 REGISTER_NAME = "DIG_Input"
318 />
319 <API_REGISTER_ALIAS NAME = "OUTPUT_LSB_REG"
320 SOURCE = "TIMER16_LSB"
321 REGISTER_NAME = "DIG_Output"
322 />
323 <API_REGISTER_ALIAS NAME = "OUTPUT_MSB_REG"
324 SOURCE = "TIMER16_MSB"
325 REGISTER_NAME = "DIG_Output"
326 />
327 </API_REGISTER_ALIAS_LIST>
328
329 <XGUI_LIST>
330 <XGUI NAME="TIMER16X_WIZARD" DISPLAY_NAME="TIMER16X Wizard..." SB
331 <XGUI NAME = "PlaceScript" SRC = "..\scripts\EventHandlers.js" TY
336 <XGUI NAME="CodeGenScript" SRC="..\scripts\EventHandlers.js" TYPE
341 <XGUI NAME = "UnPlaceScript" SRC = "..\scripts\EventHandlers.js"
342 <XGUI_FUNC_LIST>
343 <XGUI_FUNC FUNC_NAME="Timer16xUnPlaceUM" />
344 </XGUI_FUNC_LIST>
345 </XGUI>
346 </XGUI_LIST>
347
348 <API_FILE_LIST>
349 <API_FILE NAME = "Timer16INT.asm"
350 API_PREFIX = "Timer16"
351 API_PATH = "\"
352 />
353 <API_FILE NAME = "Tmr1627.inc" />
354 <API_FILE NAME = "Tmr1627.asm" />
355 <API_FILE NAME = "Tmr1627.h" />
356 </API_FILE_LIST>
357
358 </USER_MODULE>
359 </USER_MODULE_LIST>
360 </PSOC_DEVICE_DB>
361

```

5. CRET.

3.7.2 Step 24: Implement Required Functionality in Un-placement JS

1. Go to **scripts** folder and open *EventHandlers.js* file.
2. Add **Timer16xUnPlaceUM** function implementation.

Be sure that the un-placement function has no argument. The example of code which display message is represented in [Appendix B on page 66](#).

Only **PS** global object which is listed in [Table 3-2 on page 43](#) is available in un-placement JS. All methods from [Table 3-2 on page 43](#) and [Table 3-3 on page 44](#) may be used in this type of JS. There is one limitation - you cannot change current values or filter value list of UM parameters due to UM instance are not available in PD project at this moment.

A. Appendices



A.1 Appendix A

A.1.1 Timer16xPackage\Timer16xPackage\Timer16xPackage.cs

```
using System;
using System.Runtime.InteropServices;
using System.Windows.Forms;
using PSoCCommon.PSoCFramework;

namespace Timer16xPackage
{
    [Guid("F68431D4-FCD1-4e9a-A2FA-F59494CE753A"),
    ClassInterface(ClassInterfaceType.None)]
    public class Timer16xPackage:IPSoCDriverPackage
    {
        private string _instanceName; // Name of UM instance
        private string _packageType; // "PACKAGE:PROPERTY"
        private string _cmxID; // project ID
        private object _lock = new object(); // lock object
        protected string _installPath = ""; // Path to UM folder

        #region IPSoCDriverPackage Members

        public string InstallPath
        {
            get { return _installPath; }
            set { _installPath = value; }
        }

        public void OnMessage(object sender, PSoCEventArgs e)
        {
        }

        public void OnQuit(object sender, PSoCEventArgs e)
        {
        }

        public event EventHandler<PSoCEventArgs> PropertiesUpdated;

        public void SetCmxID(string cmxID)
        {
            _cmxID = cmxID;
        }
    }
}
```

```

public void SetInstanceName(string instanceName)
{
    _instanceName = instanceName;
}

public void SetPackageType(string packageType)
{
    _packageType = packageType;
}

public void Start()
{
    if (_packageType == "PACKAGE:PROPERTY")
    {
        DoWizard();
    }
}

#endregion

/// <summary>
/// cause UM parameters updating
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void OnPropertiesUpdated(object sender, EventArgs e)
{
    EventHandler<PSoCEventArgs> handler;
    lock (_lock)
    {
        handler = PropertiesUpdated;
    }
    if (handler != null)
    {
        PSoCEventArgs eventArgs = new PSoCEventArgs();
        eventArgs.Data = _instanceName;
        handler(this, eventArgs);
    }
}

/// <summary>
/// launch Wizard Form
/// </summary>
private void DoWizard()
{
    string sOverlayName = CMXEngineAccessor.GetData(_cmxID,
"GET_OVERLAY_NAME_FROM_INSTANCE", _instanceName, "", "");
    WizardForm wizardForm = new WizardForm(_cmxID, _instanceName, sOverlayName);
    if (wizardForm.ShowDialog() == DialogResult.OK)
    {
        OnPropertiesUpdated(this, null);
    }
}
}

```


A.1.2 Timer16xPackage\Timer16xPackage\WizardForm.Designer.cs

```
namespace Timer16xPackage
{
    partial class WizardForm
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;
        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise,
        false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }
        #region Windows Form Designer generated code
        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.button_OK = new System.Windows.Forms.Button();
            this.grB_Parameters = new System.Windows.Forms.GroupBox();
            this.label_InputClockValue = new System.Windows.Forms.Label();
            this.label_InputClock = new System.Windows.Forms.Label();
            this.label_TimerOutClockValue = new System.Windows.Forms.Label();
            this.label_TimerOutClock = new System.Windows.Forms.Label();
            this.label_Clock = new System.Windows.Forms.Label();
            this.cmB_Clock = new System.Windows.Forms.ComboBox();
            this.label_CompareType = new System.Windows.Forms.Label();
            this.cmB_CompareType = new System.Windows.Forms.ComboBox();
            this.label_Period = new System.Windows.Forms.Label();
            this.label_CompareValue = new System.Windows.Forms.Label();
            this.numUD_Period = new System.Windows.Forms.NumericUpDown();
            this.numUD_CompareValue = new System.Windows.Forms.NumericUpDown();
            this.grB_Parameters.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.numUD_Period)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.numUD_CompareValue)).Begini-
nInit();

            this.SuspendLayout();
            //
            // button_OK
            //
            this.button_OK.Location = new System.Drawing.Point(184, 230);
            this.button_OK.Name = "button_OK";
            this.button_OK.Size = new System.Drawing.Size(75, 23);
            this.button_OK.TabIndex = 0;
            this.button_OK.Text = "OK";
            this.button_OK.UseVisualStyleBackColor = true;
        }
    }
}
```

```

this.button_OK.Click += new System.EventHandler(this.button_OK_Click);
//
// grB_Parameters
//
this.grB_Parameters.Controls.Add(this.numUD_CompareValue);
this.grB_Parameters.Controls.Add(this.numUD_Period);
this.grB_Parameters.Controls.Add(this.label_CompareValue);
this.grB_Parameters.Controls.Add(this.label_Period);
this.grB_Parameters.Controls.Add(this.cmB_CompareType);
this.grB_Parameters.Controls.Add(this.label_CompareType);
this.grB_Parameters.Controls.Add(this.label_InputClockValue);
this.grB_Parameters.Controls.Add(this.label_InputClock);
this.grB_Parameters.Controls.Add(this.label_TimerOutClockValue);
this.grB_Parameters.Controls.Add(this.label_TimerOutClock);
this.grB_Parameters.Controls.Add(this.label_Clock);
this.grB_Parameters.Controls.Add(this.cmB_Clock);
this.grB_Parameters.Location = new System.Drawing.Point(12, 12);
this.grB_Parameters.Name = "grB_Parameters";
this.grB_Parameters.Size = new System.Drawing.Size(247, 212);
this.grB_Parameters.TabIndex = 1;
this.grB_Parameters.TabStop = false;
this.grB_Parameters.Text = "Parameters";
//
// label_InputClockValue
//
this.label_InputClockValue.AutoSize = true;
this.label_InputClockValue.Location = new System.Drawing.Point(110, 58);
this.label_InputClockValue.Name = "label_InputClockValue";
this.label_InputClockValue.Size = new System.Drawing.Size(19, 13);
this.label_InputClockValue.TabIndex = 5;
this.label_InputClockValue.Text = "----";
//
// label_InputClock
//
this.label_InputClock.AutoSize = true;
this.label_InputClock.Location = new System.Drawing.Point(43, 58);
this.label_InputClock.Name = "label_InputClock";
this.label_InputClock.Size = new System.Drawing.Size(61, 13);
this.label_InputClock.TabIndex = 4;
this.label_InputClock.Text = "Input Clock";
//
// label_TimerOutClockValue
//
this.label_TimerOutClockValue.AutoSize = true;
this.label_TimerOutClockValue.Location = new System.Drawing.Point(110, 89);
this.label_TimerOutClockValue.Name = "label_TimerOutClockValue";
this.label_TimerOutClockValue.Size = new System.Drawing.Size(19, 13);
this.label_TimerOutClockValue.TabIndex = 3;
this.label_TimerOutClockValue.Text = "----";
//
// label_TimerOutClock
//
this.label_TimerOutClock.AutoSize = true;
this.label_TimerOutClock.Location = new System.Drawing.Point(6, 89);
this.label_TimerOutClock.Name = "label_TimerOutClock";
this.label_TimerOutClock.Size = new System.Drawing.Size(98, 13);
this.label_TimerOutClock.TabIndex = 2;
this.label_TimerOutClock.Text = "Timer Output Clock";

```

```
//
// label_Clock
//
this.label_Clock.AutoSize = true;
this.label_Clock.Location = new System.Drawing.Point(70, 27);
this.label_Clock.Name = "label_Clock";
this.label_Clock.Size = new System.Drawing.Size(34, 13);
this.label_Clock.TabIndex = 1;
this.label_Clock.Text = "Clock";
//
// cmB_Clock
//
this.cmB_Clock.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
this.cmB_Clock.FormattingEnabled = true;
this.cmB_Clock.Location = new System.Drawing.Point(110, 24);
this.cmB_Clock.Name = "cmB_Clock";
this.cmB_Clock.Size = new System.Drawing.Size(121, 21);
this.cmB_Clock.TabIndex = 0;
this.cmB_Clock.SelectedIndexChanged += new System.EventHandler(
dler(this.cmB_Clock_SelectedIndexChanged));
//
// label_CompareType
//
this.label_CompareType.AutoSize = true;
this.label_CompareType.Location = new System.Drawing.Point(28, 117);
this.label_CompareType.Name = "label_CompareType";
this.label_CompareType.Size = new System.Drawing.Size(76, 13);
this.label_CompareType.TabIndex = 6;
this.label_CompareType.Text = "Compare Type";
//
// cmB_CompareType
//
this.cmB_CompareType.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDown-
List;

this.cmB_CompareType.FormattingEnabled = true;
this.cmB_CompareType.Location = new System.Drawing.Point(113, 114);
this.cmB_CompareType.Name = "cmB_CompareType";
this.cmB_CompareType.Size = new System.Drawing.Size(121, 21);
this.cmB_CompareType.TabIndex = 7;
//
// label_Period
//
this.label_Period.AutoSize = true;
this.label_Period.Location = new System.Drawing.Point(67, 146);
this.label_Period.Name = "label_Period";
this.label_Period.Size = new System.Drawing.Size(37, 13);
this.label_Period.TabIndex = 8;
this.label_Period.Text = "Period";
//
// label_CompareValue
//
this.label_CompareValue.AutoSize = true;
this.label_CompareValue.Location = new System.Drawing.Point(25, 177);
this.label_CompareValue.Name = "label_CompareValue";
this.label_CompareValue.Size = new System.Drawing.Size(79, 13);
this.label_CompareValue.TabIndex = 9;
this.label_CompareValue.Text = "Compare Value";
//
```

```

        // numUD_Period
        //
        this.numUD_Period.Location = new System.Drawing.Point(113, 144);
        this.numUD_Period.Name = "numUD_Period";
        this.numUD_Period.Size = new System.Drawing.Size(120, 20);
        this.numUD_Period.TabIndex = 10;
        this.numUD_Period.ValueChanged += new System.EventHan-
dler(this.numUD_Period_ValueChanged);
        //
        // numUD_CompareValue
        //
        this.numUD_CompareValue.Location = new System.Drawing.Point(113, 175);
        this.numUD_CompareValue.Name = "numUD_CompareValue";
        this.numUD_CompareValue.Size = new System.Drawing.Size(120, 20);
        this.numUD_CompareValue.TabIndex = 11;
        //
        // WizardForm
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(267, 261);
        this.Controls.Add(this.grB_Parameters);
        this.Controls.Add(this.button_OK);
        this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.Fixed3D;
        this.MaximizeBox = false;
        this.MinimizeBox = false;
        this.Name = "WizardForm";
        this.StartPosition = System.Windows.Forms.FormStartPosition.CenterParent;
        this.Text = "Timer16X Wizard";
        this.Load += new System.EventHandler(this.WizardForm_Load);
        this.grB_Parameters.ResumeLayout(false);
        this.grB_Parameters.PerformLayout();
        ((System.ComponentModel.ISupportInitialize)(this.numUD_Period)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.numUD_CompareValue)).EndInit();
        this.ResumeLayout(false);
    }
}
#endregion
private System.Windows.Forms.Button button_OK;
private System.Windows.Forms.GroupBox grB_Parameters;
private System.Windows.Forms.Label label_InputClockValue;
private System.Windows.Forms.Label label_InputClock;
private System.Windows.Forms.Label label_TimerOutClockValue;
private System.Windows.Forms.Label label_TimerOutClock;
private System.Windows.Forms.Label label_Clock;
private System.Windows.Forms.ComboBox cmB_Clock;
private System.Windows.Forms.NumericUpDown numUD_CompareValue;
private System.Windows.Forms.NumericUpDown numUD_Period;
private System.Windows.Forms.Label label_CompareValue;
private System.Windows.Forms.Label label_Period;
private System.Windows.Forms.ComboBox cmB_CompareType;
private System.Windows.Forms.Label label_CompareType;
}
}

```

A.1.3 Timer16xPackage\Timer16xPackage\WizardForm.cs

```
using System;
using System.Text.RegularExpressions;
using System.Windows.Forms;
using System.Xml;
using PSoCCommon.PSoCFramework;

namespace Timer16xPackage
{
    public partial class WizardForm : Form
    {
        #region Private Vars
        private string _cmxID;           // project ID
        private string _instanceName;    // Name of UM instance
        private string _overlayName;     // project name
        private XmlDocument _xmlDoc = new XmlDocument();
        #endregion Private Vars

        #region Instructor/Destructor
        public WizardForm()
        {
            InitializeComponent();
        }

        public WizardForm(string cmxId, string iName, string sOverlayName)
        {
            InitializeComponent();
            _instanceName = iName; //set instance name of User Module
            _cmxID = cmxId; //set ID of CMX engine
            _overlayName = sOverlayName;
        }
        #endregion Instructor/Destructor

        #region Handlers
        private void WizardForm_Load(object sender, EventArgs e)
        {
            string sUmParamsDom = CMXEngineAccessor.GetData(_cmxID,
"GET_USERMODULE_PARAMETERS", _overlayName, _instanceName, "");
            _xmlDoc.LoadXml(sUmParamsDom);

            //Initialize Clock combobox with values of 'Clock' UM parameter
            InitializeComboBox(cmB_Clock, "Clock");
            if (cmB_Clock.SelectedItem != null)
                SetInputClock();

            //Initialize CompareType combobox with values of 'CompareType' UM parameter
            InitializeComboBox(cmB_CompareType, "CompareType");

            //Initialize Period numeric up down control with values of 'Period' UM parameter
            InitializeNumericUpDown(numUD_Period, "Period");
            //Initialize CompareValue numeric up down control with values of 'CompareValue'
            UM parameter
            InitializeNumericUpDown(numUD_CompareValue, "CompareValue");
        }

        private void button_OK_Click(object sender, EventArgs e)
        {

```

```

    {
        //set User Module Parameters
        CMXEngineAccessor.GetData(_cmxID, "SET_USERMODULE_PARAM_NO_OVERLAY",
        _instanceName, "Clock", cmB_Clock.SelectedItem.ToString());
        CMXEngineAccessor.GetData(_cmxID, "SET_USERMODULE_PARAM_NO_OVERLAY",
        _instanceName, "CompareType", cmB_CompareType.SelectedItem.ToString());
        CMXEngineAccessor.GetData(_cmxID, "SET_USERMODULE_PARAM_NO_OVERLAY",
        _instanceName, "Period", numUD_Period.Value.ToString());
        CMXEngineAccessor.GetData(_cmxID, "SET_USERMODULE_PARAM_NO_OVERLAY",
        _instanceName, "CompareValue", numUD_CompareValue.Value.ToString());
        CMXEngineAccessor.GetData(_cmxID, "SET_USERMODULE_PARAM_NO_OVERLAY", _instanceName, "TimeLastModified",
        DateTime.Now.ToShortDateString() + " " + DateTime.Now.ToShortTimeString());
        DialogResult = DialogResult.OK;
        Close();
    }

    private void cmB_Clock_SelectedIndexChanged(object sender, EventArgs e)
    {
        SetInputClock();
        SetOutputClock();
    }

    private void numUD_Period_ValueChanged(object sender, EventArgs e)
    {
        SetOutputClock();
    }
#endregion Handlers

#region Private Methods
/// <summary>
/// Initialize combobox with values of corresponding User Module parameters
/// </summary>
/// <param name="cmb">Instance of Combobox on Wizard</param>
/// <param name="sParameterName">User Module parameter name</param>
private void InitializeComboBox(ComboBox cmb, string sParameterName)
{
    //value list
    XmlNodeList propValues = _xmlDoc.SelectNodes("//CMX_PROPERTY[@NAME=\"" + sParameterName + "\"]//CMX_PROPERTY_VALUE_LIST//CMX_PROPERTY_VALUE");
    if (propValues != null)
    {
        for (int i = 0; i < propValues.Count; i++)
        {
            cmb.Items.Add((propValues[i] as XmlElement).GetAttribute("NAME"));
        }
    }

    //current value
    XmlElement propElem = _xmlDoc.SelectSingleNode("//CMX_PROPERTY[@NAME=\"" + sParameterName + "\"] as XmlElement);
    if (propElem != null)
    {
        cmb.SelectedItem = propElem.GetAttribute("CURRENT_VALUE");
    }
}

private void InitializeNumericUpDown (NumericUpDown numericUpDown, string sParameterName)
{

```

```

        XmlElement propElem = _xmlDoc.SelectSingleNode("//CMX_PROPERTY[@NAME=\"\" + sPa-
parameterName + "\"]") as XmlElement;
        if (propElem != null)
        {
            string sTemp = propElem.GetAttribute("MIN");
            numericUpDown.Minimum = Int32.Parse(sTemp);
            sTemp = propElem.GetAttribute("MAX");
            numericUpDown.Maximum = Int32.Parse(sTemp);
            sTemp = propElem.GetAttribute("CURRENT_VALUE");
            numericUpDown.Value = Int32.Parse(sTemp);
        }
    }

    /// <summary>
    /// Get current value of global resource
    /// </summary>
    /// <param name="resourceName">Global Resource Name</param>
    /// <returns>Current value of global resource</returns>
    private string GetGlobalResourceValue(string resourceName)
    {
        string sGlobalResources = CMXEngineAccessor.GetData(_cmxID,
"GET_GLOBAL_RESOURCES", _overlayName, _instanceName, "");
        XmlDocument xmlDoc = new XmlDocument();
        xmlDoc.LoadXml(sGlobalResources); // Load global resource DOM
        XmlElement resourceElem = xmlDoc.SelectSingleNode("//CMX_PROPERTY[@NAME=\"\" +
resourceName + "\"]") as XmlElement;
        if (resourceElem != null)
            return resourceElem.GetAttribute("CURRENT_VALUE");
        return null;
    }

    /// <summary>
    /// Set Input Clock value
    /// This value depends on 'Clock' UM parameter and
    /// values of global resource
    /// </summary>
    private void SetInputClock()
    {
        long lInputClockValue;
        //get current value of 'Clock' parameter
        string sClockValue = cmB_Clock.SelectedItem.ToString();
        //get value of 'Power Setting [ Vcc / SysClk freq ]'
        string sSysClkValue = GetGlobalResourceValue("Power Setting [ Vcc / SysClk freq
]");

        sSysClkValue = sSysClkValue.Substring(sSysClkValue.IndexOf('/') + 1);
        sSysClkValue = Regex.Match(sSysClkValue, @"\d+").Value;
        long lSysClkValue = Int32.Parse(sSysClkValue) * 1000000; //convert to Hz

        //get value of 'VC1= SysClk/N'
        string sVC1Value = GetGlobalResourceValue("VC1= SysClk/N");
        long lVC1Value = lSysClkValue / long.Parse(sVC1Value);

        //get value of 'VC2= VC1/N'
        string sVC2Value = GetGlobalResourceValue("VC2= VC1/N");
        long lVC2Value = lVC1Value / long.Parse(sVC2Value);

        //get value of 'VC3'
        string sVC3Source = GetGlobalResourceValue("VC3 Source");

```

```

string sVC3Divider = GetGlobalResourceValue("VC3 Divider");
long lVC3Value;
//calculate VC3 value: VC3 Source/VC3 Divider
switch (sVC3Source)
{
    case "SysClk/1":
        lVC3Value = lSysClkValue / Int16.Parse(sVC3Divider);
        break;
    case "VC1":
        lVC3Value = lVC1Value / Int16.Parse(sVC3Divider);
        break;
    case "SysClk*2":
        lVC3Value = lSysClkValue * 2 / Int16.Parse(sVC3Divider);
        break;
    default: //VC2
        lVC3Value = lVC2Value / Int16.Parse(sVC3Divider);
        break;
}
//calculate value of Input Clock
switch (sClockValue)
{
    case "SysClk*2":
        lInputClockValue = lSysClkValue * 2;
        break;
    case "VC1":
        lInputClockValue = lVC1Value;
        break;
    case "VC2":
        lInputClockValue = lVC2Value;
        break;
    case "VC3":
        lInputClockValue = lVC3Value;
        break;
    case "CPU_32_KHz":
        lInputClockValue = 32000;
        break;
    default:
        lInputClockValue = -1;
        break;
}
//display value of Input Clock
if (lInputClockValue == -1)
    label_InputClockValue.Text = "----";
else label_InputClockValue.Text = lInputClockValue + " Hz";
}

/// <summary>
/// Set value of Output Clock
/// Output Clock = Input Clock/(Period+1)
/// </summary>
private void SetOutputClock()
{
    if (label_InputClockValue.Text != "----")
    {
        //get value of Input Clock
        string sInputClock = Regex.Match(label_InputClockValue.Text, @"\d+").Value;
        //convert string to long
        long lInputClock = long.Parse(sInputClock);
    }
}

```



```
        //get value of Period
        long lPeriod = long.Parse(numUD_Period.Value.ToString());
        //calculate Output Clock
        long lOutputClock = lInputClock/(lPeriod + 1);
        //Display Output Clock
        label_TimerOutClockValue.Text = lOutputClock + " Hz";
    }
}

#endregion Private Methods
}
}
```

A.2 Appendix B

A.2.1 Timer16X\Ver_2_6\scripts\EventHandlers.js

```

/*
This function is calling before CompareOut
value parameter changing
*/
function CompareOutSelect(strParam)
{
    var devEditor = PS.GetDeviceEditor();
    var xmlDOM = new ActiveXObject("Microsoft.XMLDOM");
    xmlDOM.validateOnParse = false;
    xmlDOM.setProperty("SelectionLanguage","XPath");
    xmlDOM.loadXML(strParam);

    var sInstName = xmlDOM.documentElement.getAttribute("UM_INST_NAME");
    var sOverlay = devEditor.GetData("GET_OVERLAY_NAME_FROM_INSTANCE", sInstName, "", "",
    "");

    var sTerminalCountOut = devEditor.GetUserModuleParameter2(sOverlay, sInstName, "Termi-
    nalCountOut");//get current value
    var sCompareOutSettingsArray = otto.GetCurUMParamValueRange("CompareOut").split(",");//
    get values and compose array from them
    //Remove the value that is being used by the TerminalCountOut UM parameter from the list
    for (var i = 0; i < sCompareOutSettingsArray.length; i++)
    {
        if (sCompareOutSettingsArray[i] == sTerminalCountOut)
        {
            sCompareOutSettingsArray.splice(i, 1);
        }
    }
    //Generate new list with the value, being use by TerminalCountOut, taken out
    otto.SetCurUMParamValueRange("CompareOut", sCompareOutSettingsArray.join());
}

/*configure current value of 'TC_PulseWidth' UM parameter
during User Module placement
if value of 'SysClk*2 Disable' global parameter is 'Yes'
'TC_PulseWidth' parameter should have 'Full Clock', otherwise - 'One-Half Clock'
*/
function Timer16xPlaceUM(sInstName)
{
    var devEditor = PS.GetDeviceEditor();
    var sOverlay = devEditor.GetData("GET_OVERLAY_NAME_FROM_INSTANCE", sInstName, "", "",
    ""); //get project ID
    var sSysclk = devEditor.GetGlobalResourceValue(sOverlay, "SysClk*2 Disable");//get value
    of 'SysClk*2 Disable' parameter

    if(sSysclk=='No')
        devEditor.SetUserModuleParameter2(sOverlay, sInstName, "TC_PulseWidth", "One-Half
        Clock");//set current value
    }

    /*set current value of 'CompareValue' UM parameter

```

```
to 'Period' parameter value
if 'CompareValue' is great then 'Period'
during project generation
*/
function Timer16xGenerateCode(sInstName)
{
    var devEditor = PS.GetDeviceEditor();
    var sOverlay = devEditor.GetData("GET_OVERLAY_NAME_FROM_INSTANCE", sInstName, "", "",
    ""); //get project ID

    var sPeriod = devEditor.GetUserModuleParameter2(sOverlay, sInstName, "Period");//get
current value
    var sCompareValue = devEditor.GetUserModuleParameter2(sOverlay, sInstName, "Compare-
Value");//get current value

    if(parseInt(sCompareValue)>parseInt(sPeriod))//if CompareValue > Period
        //set current value of CompareValue to Period
        devEditor.SetUserModuleParameter2(sOverlay, sInstName, "CompareValue", sPeriod);
}

/*Show a dialog when the UM is removed
*/
function Timer16xUnPlaceUM()
{
    PS.ShowMessageBox("Timer16X Custom User Module was removed from current project.")
}
```

A.3 Appendix C

A.3.1 Timer16X\Ver_2_6\scripts\Timer16xRules.js

```
function CheckTimerOutputClock(sInstName)
{
    var devEditor = PS.GetDeviceEditor();
    var sOverlay = devEditor.GetData("GET_OVERLAY_NAME_FROM_INSTANCE", sInstName, "", "",
    "");

    //get current value of 'Clock' parameter
    var sClockValue = devEditor.GetUserModuleParameter2(sOverlay, sInstName, "Clock");
    var sPeriodValue = devEditor.GetUserModuleParameter2(sOverlay, sInstName, "Period");

    //get value of 'Power Setting [ Vcc / SysClk freq ]'
    var sSysClkValue = devEditor.GetGlobalResourceValue(sOverlay, "Power Setting [ Vcc / Sys-
    Clk freq ]");
    var lSysClkValue;
    switch(sSysClkValue)
    {
        case "5.0V / 24MHz":
        case "3.3V / 24MHz": lSysClkValue = "24000000"; break;
        case "2.7V / 12MHz": lSysClkValue = "12000000"; break;
        case "5.0V / 6MHz" :
        case "3.3V / 6MHz" :
        case "2.7V / 6MHz" : lSysClkValue = "6000000"; break;
        default: lSysClkValue = "24000000"; break;
    }

    //get value of 'VC1= SysClk/N'
    var sVC1Value = devEditor.GetGlobalResourceValue(sOverlay, "VC1= SysClk/N");
    var lVC1Value = lSysClkValue/parseInt(sVC1Value);
    //get value of 'VC2= VC1/N'
    var sVC2Value = devEditor.GetGlobalResourceValue(sOverlay, "VC2= VC1/N");
    var lVC2Value = lVC1Value/parseInt(sVC2Value);
    //get value of 'VC3'
    var sVC3Source = devEditor.GetGlobalResourceValue(sOverlay, "VC3 Source");
    var sVC3Divider = devEditor.GetGlobalResourceValue(sOverlay, "VC3 Divider");
    //calculate VC3 value: VC3 Source/VC3 Divider
    switch (sVC3Source)
    {
        case "SysClk/1":
            lVC3Value = lSysClkValue/parseInt(sVC3Divider);
            break;
        case "VC1":
            lVC3Value = lVC1Value/parseInt(sVC3Divider);
            break;
        case "SysClk*2":
            lVC3Value = lSysClkValue * 2/parseInt(sVC3Divider);
            break;
        default: //VC2
            lVC3Value = lVC2Value/parseInt(sVC3Divider);
            break;
    }
    //calculate value of Input Clock
    switch (sClockValue)
```

```
{
    case "SysClk*2":
        lInputClockValue = lSysClkValue * 2;
        break;
    case "VC1":
        lInputClockValue = lVC1Value;
        break;
    case "VC2":
        lInputClockValue = lVC2Value;
        break;
    case "VC3":
        lInputClockValue = lVC3Value;
        break;
    case "CPU_32_KHz":
        lInputClockValue = 32000;
        break;
    default:
        lInputClockValue = -1;
        break;
}

var lTimerOutputClock;
if(lInputClockValue!=-1)
    lTimerOutputClock = lInputClockValue/(parseInt(sPeriodValue)+1);
if(lTimerOutputClock>8000000)
    ShowWarningMsg("Timer Output Clock for " + sInstName + " is greater than 8 MHz. " +
    " That is why " + sInstName +" can not be connected to internal source.");
}
```

