1. Validácia návrhu z príkazového riadku

1.1. Spusťte Vivado HLS cez **Start > Všetky programy > Xilinx Design Tools > Vivado HLS 2015.4 Command Prompt**

1.2. V spustenom príkazovom riadku zmeňte svoje umiestnenie príkazom **cd c:\Vivado_projects\lab3.**

1.3. Vlastné kontroly programu (dct_test.c) máme k dispozícii. Jeho použitím môžeme návrh overiť. Taktiež je k dispozícii Makefile. Použitím súboru Makefile sa potrebné zdrojové súbory skompilujú a skompilovaný program môžeme spustiť. V príkazovom riadku Vivado HLS, príkazom **make** spustíte kompiláciu. Všimnite si, že zdrojové súbory (dct.c a dct_test.c) sa skompilovali, bol vytvorený dct spustiteľný program a následne bol spustený. Program testuje návrh a ako výstup vypísal správu Výsledok je správny (Results are good).

1.4. Vypnite príkazový riadok príkazom **exit**.

2. Vytvorenie nového projektu

2.1. Spuťte Vivado HLS: vyberte **Start > Všetky programy > Xilinx Design Tools > Vivado HLS 2015.4**

2.2. V spustenom programe, vyberte **File > New Project.** Otvorí sa sprievodca vytvorenia nového projektu.

2.3. Kliknite na **Browse...** a ako cieľ uloženia projektu vyberte c:\Vivado_projects\lab3 a kliknite na **OK.**

2.4. Nazvite projekt **dct.prj**.

2.5. Kliknite na **Next**.

2.6. V okne *Add/Remove Files* ako názov funkcie (časť Top Function) napíšte **dct** (za predpokladu, že zdrojový súbor funkcie, sa nazýva dct).

2.7. Kliknite na **Add Files...**, vyberte súbor *dct.c* z **c:\Vivado_projects\lab3** a kliknite na **Open**.

2.8. Kliknite na **Next**.

2.9. V okne *Add/Remove Files* pre testovanie návrhu, kliknite na **Add Files...**, vyberte súbory *dct_test.c, in.dat, out.golden.dat* **c:\Vivado_projects\lab3** a kliknite na **Open**.

2.10. Kliknite na **Next**.

2.11. V okne *Solution Configuration*, nechajte názov riešenia **solution1** a zmeňte taktovací cyklus na **10** (pre ZedBoard) alebo **8** (pre Zybo). Riadok nespoľahlivosti (Uncertainty) nechajte prázdny (ZedBoard si stanoví predvolenú hodnotu na 1.25 a Zybo na 1).

2.12. Kliknite na **...** v časti Pa rt Selection, a využite filter a vyberte súčasť **xc7z020clg484-1** (ZedBoard) alebo **xc7z010clg400-1** (Zybo) a kliknite na **OK:**
Family: **Zynq**
Sub-Family: **Zynq**
Package: **clg484** (ZedBoard) or **clg400** (Zybo)
Speed Grade: **–1**

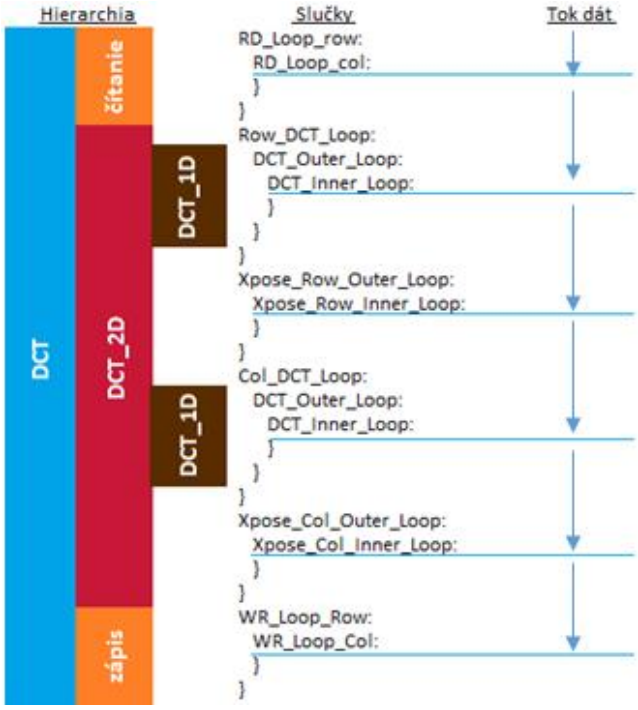
2.13. Kliknite na **Finish**.

2.14. Dvakrát kliknite na **dct.c** pod priečinkom *source* a jeho obsah sa otvorí v informačnom okne.

```
78 void dct(short input[N], short output[N])
79 {
80
81    short buf_2d_in[DCT_SIZE][DCT_SIZE];
82    short buf_2d_out[DCT_SIZE][DCT_SIZE];
83
84    // Read input data. Fill the internal buffer.
85    read_data(input, buf_2d_in);
86
87    dct_2d(buf_2d_in, buf_2d_out);
88
89    // Write out the results.
90    write_data(buf_2d_out, output);
91 }
```

Obrázok 1 Posudzovaný návrh

Funkcia najvyššej úrovne **dct**, je definovaná v riadku 78. Je tu implementovaný 2D DCT algoritmus, ktorý ako prvé spracováva každý riadok na vstupe poľa cez 1D DCT a potom spracováva stĺpce výsledného poľa cez rovnaký 1D DCT. Volajú to funkcie read_data, dct_2d a write_data. Funkcia **read_data** je definovaná na riadku 54 a pozostáva z dvoch slučiek – RD_Loop_Row a RD_Loop_Col. Funkcia **write_data** je definovaná na riadku 66 a pozostáva z dvoch slučiek vykonávajúce zapisovanie výsledku. Funkcia **dct_2d** je definovaná na riadku 23, volá funkciu dct_1d a vykonáva prenos. Napokon, funkcia **dct_1d** je definovaná na riadku 4, používa dct_coeff_table a vykonáva požadovanú funkciu zavedením základných iteračných foriem z 1D Type-II DCT algoritmu. Nasledujúci obrázok ukazuje hierarchiu na ľavej strane, slučky v poradí ako sa vykonávajú a tok dát na pravej strane.



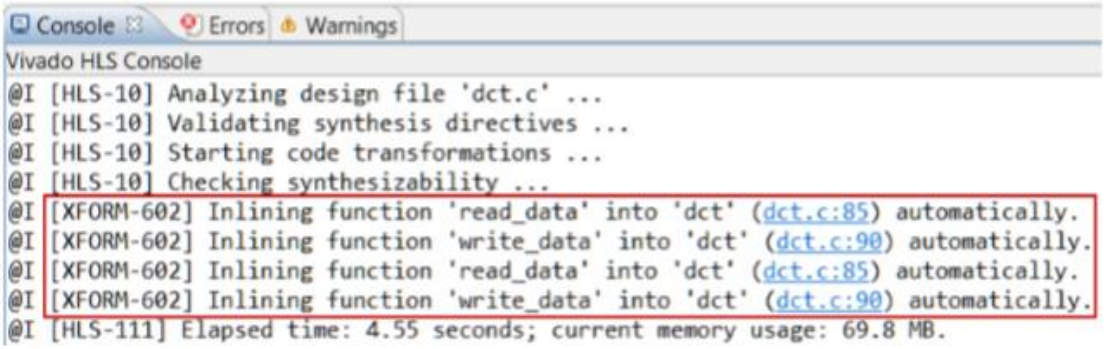
Obrázok 2 Hierarchia návrhu

3. Syntéza návrhu

3.1. Vyberte **Solution > Run C Synthesis > Active Solution** alebo kliknite na ▶ pre spustenie syntézy.

3.2. Po dokončení syntézy, správa viacerých súborov bude k dispozícii a výsledok syntézy sa zobrazí v informačnom okne.

Všimnite si, že v súhrnnej správe v okne Explorer je možné vidieť len položky dct_1d.rpt, dct_2d.rpt a dct.rpt. Správa funkcií write_data a read_data nie sú uvedené. Je to tak, pretože tieto dve funkcie boli vnorené. Môžete si to overiť posúvaním sa v konzolovom okne Vivado HLS.



Obrázok 3 Inlining funkcii read_data a write_data

3.3. Výsledná správa ukazuje odhad výkonu a zdrojov, ako aj odhad latencie v návrhu. Všimnite si, že návrh nie je optimalizovaný ani prepojený.

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	6.38	1.25

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
3959	3959	3960	3960	none

Detail

Instance

Loop

	Latency			Initiation Interval			
Loop Name	min	max	Iteration Latency	achieved	target	Trip Count	Pipelined
- RD_Loop_Row	144	144	18	-	-	8	no
+ RD_Loop_Col	16	16	2	-	-	8	no
- WR_Loop_Row	144	144	18	-	-	8	no
+ WR_Loop_Col	16	16	2	-	-	8	no

Obrázok 4 Správa syntézy

3.4. Preštudujte si správu a odpovedzte na nasledujúce otázky:

Odhadovaná perióda taktu: _____

Najhorší prípad latencie: _____

Celková hodnota DSP48E: _____

Celková hodnota BRAM: _____

Celková hodnota FF: _____

Celková hodnota LUT: _____

3.5. Správa tiež ukazuje rozhranie signálov najvyššej úrovne vygenerované nástrojmi.

Interface

Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
ap_clk	in	1	ap_ctrl_hs	dct	return value
ap_rst	in	1	ap_ctrl_hs	dct	return value
ap_start	in	1	ap_ctrl_hs	dct	return value
ap_done	out	1	ap_ctrl_hs	dct	return value
ap_idle	out	1	ap_ctrl_hs	dct	return value
ap_ready	out	1	ap_ctrl_hs	dct	return value
input_r_address0	out	6	ap_memory	input_r	array
input_r_ce0	out	1	ap_memory	input_r	array
input_r_q0	in	16	ap_memory	input_r	array
output_r_address0	out	6	ap_memory	output_r	array
output_r_ce0	out	1	ap_memory	output_r	array
output_r_we0	out	1	ap_memory	output_r	array
output_r_d0	out	16	ap_memory	output_r	array

Obrázok 5 Generované rozhranie signálov

Môžete vidieť, že ap_clk, ap_rst boli automaticky pridané. Ap_start, ap_done, ap_idle a ap_ready sú signály používané na najvyššej úrovni ako prevzaté signály, ktoré indikujú, kedy je návrh schopný prijať ďalší príkaz na výpočet (ap_idle), kedy začína ďalší výpočet (ap_start) a kedy je výpočet dokončený (ap_done). Funkcia najvyššej úrovne obsahuje vstupné a výstupné polia, preto rozhranie ap_memory je generované pre každé z nich.

3.6. Otvorte súbory dct_1d.rpt a dct_2d.rpt buď cez okno Explorer alebo pomocou odkazu v dolnej časti dct.rpt v informačnom okne. Správa pre dct_2d jasne naznačuje, že väčšina periód návrhu (3 668) je použitá na vytváranie riadkov a stĺpcov DCT. Taktiež, správa dct_1d naznačuje, že latencia je 209 taktových periód ((24+2)*8+1).

4. Spustenie Co-simulácie

4.1. Vyberte **Solution > Run C/RTL Cosimulation** alebo kliknite na pre spustenie požadovanej simulácie.
Otvorí sa dialógové okno co-simulácie.

4.2. Vyberte možnosť **Verilog** a kliknite na **OK** pre spustenie Verilog simulácie, ktorá používa XSIM simulátor.
Spustí sa RTL co-simulácia, vygeneruje sa a skompiluje niekoľko súborov a potom sa nasimuluje návrh. V konzolovom okne môžete vidieť proces simulácie ako aj správu o prejdení testu (Test passed).

5. Použitie direktívy PIPELINE

5.1. **Vytvorenie nového riešenia kopírovaním nastavení predchádzajúceho riešenia. Použitie direktívy PIPELINE. Generovanie riešenia a analýza výstupu.**

5.1.1. Vyberte **Project > New Solution** alebo kliknite na na paneli nástrojov.

5.1.2. Zobrazí sa dialógové okno *Solution Configuration*. Kliknite na **Finish** (s výberom kopírovania zo Solution1).

5.1.3. Uistite sa, že je **dct.c** otvorené v informačnom okne a kliknite na kartu **Directive**.

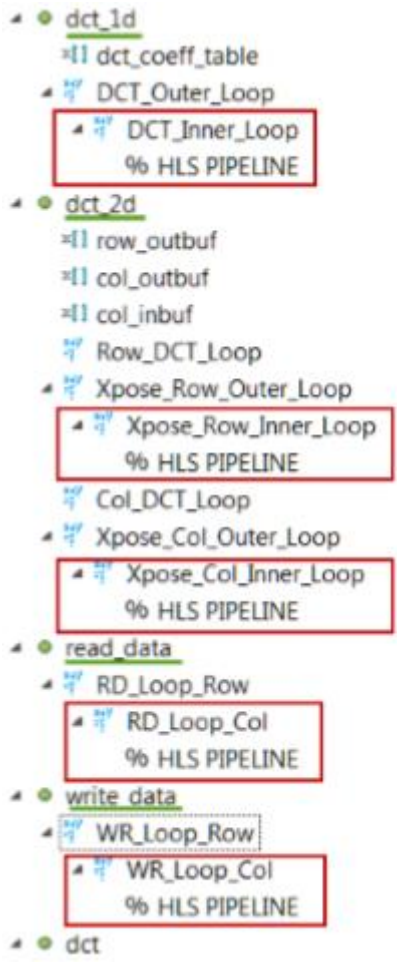
5.1.4. Kliknite pravým tlačidlom myši na **DCT_Inner_Loop** z funkcie **dct_1d** a vyberte možnosť *Insert Directive...*

5.1.5. Z kontextového menu vyberte možnosť **PIPELINE**.

5.1.6. Pole **II** (Initiation Interval) nechajte prázdne.

5.1.7. Kliknite na **OK**.

5.1.8. Rovnako použite direktívu **PIPELINE** aj pre **Xpose_Row_Inner_Loop** a **Xpose_Col_Inner_Loop** z funkcie dct_2d, **RD_Loop_Col** z funkcie read_data a **WR_Loop_Col** z funkcie write_data.



Obrázok 6 Použitie direktívy PIPELINE

5.1.9. Spusťte syntézu.

5.1.10. Keď je syntéza kompletná, vyberte **Project > Compare Reports...** alebo kliknite na  pre porovnanie dvoch riešení.

5.1.11. Vyberte *Solution1* a *Solution2* z **Available Reports**, kliknite na **Add>>** a následne kliknite na **OK**.

5.1.12. Pozorujte, že latencia sa zredukovala z 3 959 na 1 850 periód taktu (ZedBoard) a z 3 959 na 1854 (Zybo).

Performance Estimates

Timing (ns)

Clock		solution2	solution1
ap_clk	Target	10.00	10.00
	Estimated	7.68	6.38

Latency (clock cycles)

		solution2	solution1
Latency	min	1850	3959
	max	1850	3959
Interval	min	1851	3960
	max	1851	3960

a) ZedBoard

Performance Estimates

Timing (ns)

Clock		solution2	solution1
ap_clk	Target	8.00	8.00
	Estimated	6.60	6.38

Latency (clock cycles)

		solution2	solution1
Latency	min	1854	3959
	max	1854	3959
Interval	min	1855	3960
	max	1855	3960

b) Zybo

Obrázok 7 Porovnanie výkonu po pipeliningu

5.1.13. Presuňte sa nižšie v porovnávacej správe aby ste videli využitie prostriedkov. Pozorujte, že hodnota prostriedkov FF a/alebo LUT sa zvýšila zatiaľ čo BRAM a DSP48E ostali nezmenené.

Utilization Estimates

	solution2	solution1
BRAM_18K	5	5
DSP48E	1	1
FF	255	278
LUT	458	354

a) ZedBoard

Utilization Estimates

	solution2	solution1
BRAM_18K	5	5
DSP48E	1	1
FF	289	278
LUT	462	354

b) Zybo

Obrázok 8 Využitie zdrojov po pipeliningu

5.2. Otvorenie perspektívy analýzy a určenie, ktorá z taktových periód má najvyššiu latenciu.

5.2.1. Kliknite na perspektívu analýzy.

5.2.2. V okne Module Hierarchy vyberte položku dct a pozorujte položky RD_Loop_Row_RD_Loop_Col a WR_Loop_Row_WR_Loop_Col. Toto sú dve vnorené sploštené slučky a dané nové mená sú vytvorené pridaním názvu vnútornej slučky do názvu vonkajšej slučky. Môžete si to overiť v konzolovom okne prezretím správy.

5.2.3. Na karte Module Hierarchy, rozbaľte **dct > dct_2d > dct_1d**. Všimnite si, že najvyššia latencia je pri funkcii dct_2d.

5.2.4. Na karte Module Hierarchy si všimnite, že stále existuje hierarchia v dct_2d. Rozbaľte **dct > dct_2d > dct_1d** a vyberte položku *dct_1d*.

5.2.5. Na karte Performance Profile, vyberte položku DCT_Inner_Loop, kliknite pravým tlačidlom myši na blok node_60 (write) v stĺpci C3 a vyberte možnosť Goto Source. Všimnite si, že sa riadok 19 zvýraznil, ten bráni splošteniu DCT_Outer_Loop.

5.2.6. Prepnite sa na perspektívu *Synthesis*.

5.3. Vytvorenie nového riešenia kopírovaním nastavení predchádzajúceho riešenia. Použitie direktívy PIPELINE pre vonkajšie slučky. Generovanie výsledku a analýza výstupu.

5.3.1. Vyberte **Project > New Solution**.

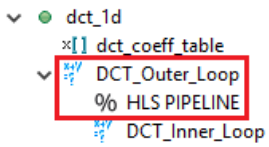
5.3.2. Otvorí sa dialógové okno *Solution Configuration*. Kliknite na **Finish** (s výberom Solution2).

5.3.3. Kliknite pravým tlačidlom myši na direktívu **PIPELINE** z **DCT_Inner_Loop** funkcie dct_1d v okne Directive a vyberte *Remove Directive.*

5.3.4. Kliknite pravým tlačidlom myši na **DCT_Outer_Loop** funkcie dct_1d a vyberte *Insert Directive...*

5.3.5. Z kontextového menu vyberte direktívu **PIPELINE**

5.3.6. Kliknite na **OK**.



Obrázok 9 Direktíva PIPELINE aplikovaná na DCT_Outer_Loop

Po zavedení direktívy na vonkajšiu slučku, všetky vnútorné slučky sa automaticky rozvinú (ak je to legálne), takže nie je potrebné explicitne použiť direktívu UNROLL

na DCT_Inner_Loop. Presunutím pipeline na vonkajšie slučky, budú vnorené slučky stále prepojené, ale operácie v tele vnútornej slučky budú pôsobiť súčasne.

5.3.7. Spusťte syntézu.

5.3.8. Po skončení syntézy, vyberte **Project > Compare Reports...** pre porovnanie dvoch riešení.

5.3.9. Pre porovnanie vyberte Solution2 a Solution3 a následne kliknite na **OK**.

5.3.10. Pozorujte, že sa latencia zredukovala z 1 850 na 874 taktových periód pre ZedBoard (z 1 854 na 878 pre Zybo).

Performance Estimates

Timing (ns)

Clock		solution3	solution2
ap_clk	Target	10.00	10.00
	Estimated	9.40	7.68

Latency (clock cycles)

		solution3	solution2
Latency	min	874	1850
	max	874	1850
Interval	min	875	1851
	max	875	1851

a) ZedBoard

Performance Estimates

Timing (ns)

Clock		solution3	solution2
ap_clk	Target	8.00	8.00
	Estimated	9.40	6.60

Latency (clock cycles)

		solution3	solution2
Latency	min	878	1854
	max	878	1854
Interval	min	879	1855
	max	879	1855

b) Zybo

Obrázok 10 Porovnanie výkonu po pipeliningu

5.3.11. Presuňte sa nižšie v porovnávacej správe aby ste videli využitie zdrojov. Pozorujte, či sa hodnoty využívania zdrojov zvýšili (okrem BRAM). Keďže DCT_Inner_Loop bol rozvinutý, paralelné výpočty vyžadujú 8 DSP48E.

Utilization Estimates

	solution3	solution2
BRAM_18K	5	5
DSP48E	8	1
FF	677	255
LUT	520	458

Utilization Estimates

	solution3	solution2
BRAM_18K	5	5
DSP48E	8	1
FF	711	289
LUT	524	462

Obrázok 11 Využitie zdrojov po pipeliningu

5.3.12. Otvorte dct_td správu a všimnite si, že počiatočný interval (Initiation Interval) predstavuje 4 periódy, nie 1 akoby sme predpokladali a hodnota BRAM je momentálne 8, ten je používaný pre tabuľku koeficientov.

Pozrite sa bližšie na záznam syntézy, všimnite si, že tabuľka koeficientov bola automaticky rozdelená, vyplýva to z 8 samostatných ROM.

Dôvodom, že počiatočný interval je 4 a nie 8 ako by sa dalo čakať je, že Vivado HLS automaticky používa dvojitý RAM port, keď je to potrebné pre plánovanie operácií.

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	9.40	1.25

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
36	36	36	36	none

Detail

Instance

Loop

	Latency			Initiation Interval			
Loop Name	min	max	Iteration Latency	achieved	target	Trip Count	Pipelined
- DCT_Outer_Loop	34	34	7	4	1	8	yes

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	8	-	-
Expression	-	-	0	128
FIFO	-	-	-	-
Instance	-	-	-	-
Memory	0	-	119	16
Multiplexer	-	-	-	21
Register	-	-	420	-
Total	0	8	539	165
Available	280	220	106400	53200
Utilization (%)	0	3	~0	~0

Obrázok 12 Zvýšenie zdrojov využitím dct_1d

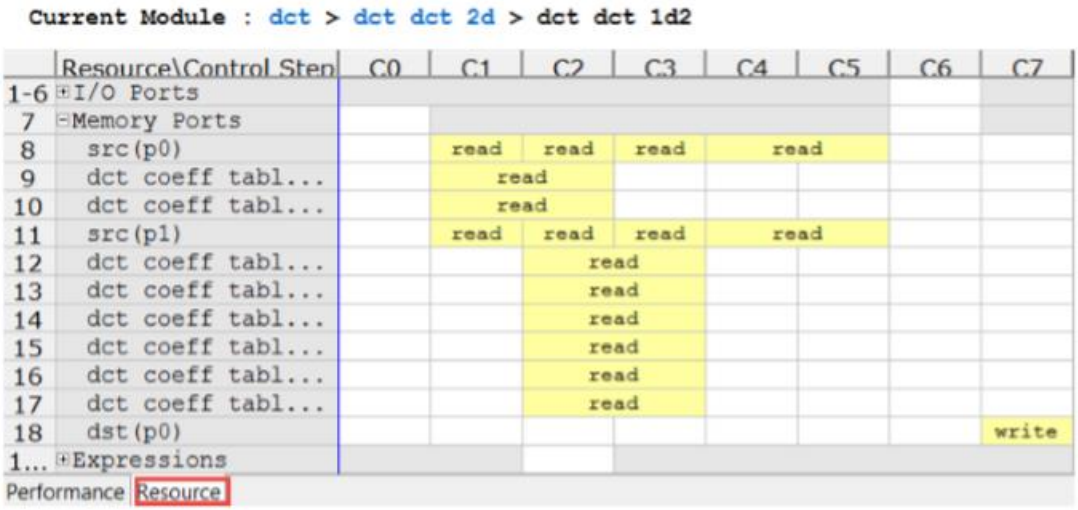
5.4. **Vykonanie analýzy návrhu prepínaním sa do perspektívy analýzy.**

5.4.1. Prepnite sa do perspektívy analýzy, rozšírte položky *Module Hierarchy* a kliknutím označte položku dct_1d.

5.4.2. Rozšírte, ak je to potrebné, kartu danej položky v profile a všimnite si, že DCT_Outer_Loop je prepojený a nenachádza sa tu DCT_Inner_Loop.

5.4.3. Vyberte položku dct_1d na karte *Module Hierarchy* a pozorujte, že DCT_Outer_Loop pokrýva viac ako 8 stavov (na karte Performance).

5.4.4. Prepnite sa na kartu Resource, rozviňte položku Memory Ports a pozorujte, že prístupová pamäť na BRAM src je používaná na maximum v každej taktovej perióde. (Nanajvýš BRAM môže byť dualport a používa oba porty). Je to dobré znamenie návrhu, že šírka pásma je ohraničená pamäťou prostriedkov.



Obrázok 13 Karta Resource

5.4.5. Prepnite sa na perspektívu syntézy.

6. Zlepšenie rozsahu pamäte

6.1. **Vytvorenie nového riešenia kopírovaním nastavení predchádzajúceho riešenia. Použitie direktívy ARRAY_PARTITION pre buf_2d_in z dct a pre col_inbuf z dct_2d. Generovanie riešenia.**

6.1.1. Vyberte **Project > New Solution** pre vytvorenie nového riešenia.

6.1.2. V zobrazenom dialógovom okne kliknite na **Finish** (s výberom solution3).

6.1.3. V otvorenom dct.c kliknite pravým tlačidlom myši na pole buf_2d_in funkcie dct na karte *Directive* a vyberte možnosť *Insert Directive...*

6.1.4. Z kontextového menu vyberte direktívu **ARRAY_PARTITION.**

6.1.5. Uistite sa, že typ je nastavený na *complete*. Dimenziu nastavte na hodnotu **2** a kliknite na **OK**.

6.1.6. Podobne nastavte direktívu *ARRAY_PARTITION* s dimenziou 2 pre **col_inbuf**.

6.1.7. Spusťte syntézu.

6.1.8. Keď je syntéza kompletná, vyberte **Project > Compare Reports...** pre porovnanie dvoch riešení.

6.1.9. Pre porovnanie vyberte *Solution3* a *Solution4*.

6.1.10. Všimnite si, že sa latencia znížila z 874 na 508 taktových periód pre ZedBoard (z 878 ba 512 pre Zybo).

Performance Estimates

Timing (ns)

Clock		solution4	solution3
ap_clk	Target	10.00	10.00
	Estimated	10.79	9.40

Latency (clock cycles)

		solution4	solution3
Latency	min	508	874
	max	508	874
Interval	min	509	875
	max	509	875

a) ZedBoard

Performance Estimates

Timing (ns)

Clock		solution4	solution3
ap_clk	Target	8.00	8.00
	Estimated	9.40	9.40

Latency (clock cycles)

		solution4	solution3
Latency	min	512	878
	max	512	878
Interval	min	513	879
	max	513	879

b) Zybo

Obrázok 14 Porovnanie výkonu po použití direktívy ARRAY_PARTITION

6.1.11. Presuňte sa nižšie v porovnávacej správe aby ste videli využitie zdrojov (Utilization Estimates). Všimnite si, že sa hodnota využitia FF zvýšila (takmer dvojnásobne).

Utilization Estimates

	solution4	solution3
BRAM_18K	3	5
DSP48E	8	8
FF	1243	677
LUT	634	520

a) ZedBoard

Utilization Estimates

	solution4	solution3
BRAM_18K	3	5
DSP48E	8	8
FF	1284	711
LUT	638	524

b) Zybo

Obrázok 15 Využitie zdrojov po použití direktívy ARRAY_PARTITION

6.2. **Vykonanie analýzy zdrojov prepnutím sa do pohľadu analýzy a prezeranie profilu zdrojov dct.**

6.2.1. Prepnite sa do pohľadu analýzy, rozbaľte položky Module Hierarchy a vyberte položku dct.

6.2.2. Prepnite sa na kartu **Resource Profile.**

6.2.3. Rozbaľte položky Memories a Expressions a všimnite si, že najviac zdrojov sú spotrebované inštanciami. Pole buf_2d_in je rozdelené na viaceré spomienky a väčšina operácií je vykonaná v súčtoch a porovnaniach.

6.2.4. Prepnite sa do perspektívy syntézy.

7. Použitie direktívy DATAFLOW

7.1. **Vytvorenie nového riešenia kopírovaním nastavení predchádzajúceho riešenia. Použitie direktívy DATAFLOW pre zlepšenie priepustnosti. Generovanie riešenia a analýza výstupu.**

7.1.1. Vyberte **Project > New Solution**.

7.1.2. V otvorenom dialógovom okne kliknite na **Finish** (s výberom solution4).

7.1.3. Zatvorte všetky neaktívne okná výberom **Project > Close Inactive Solution tabs.**

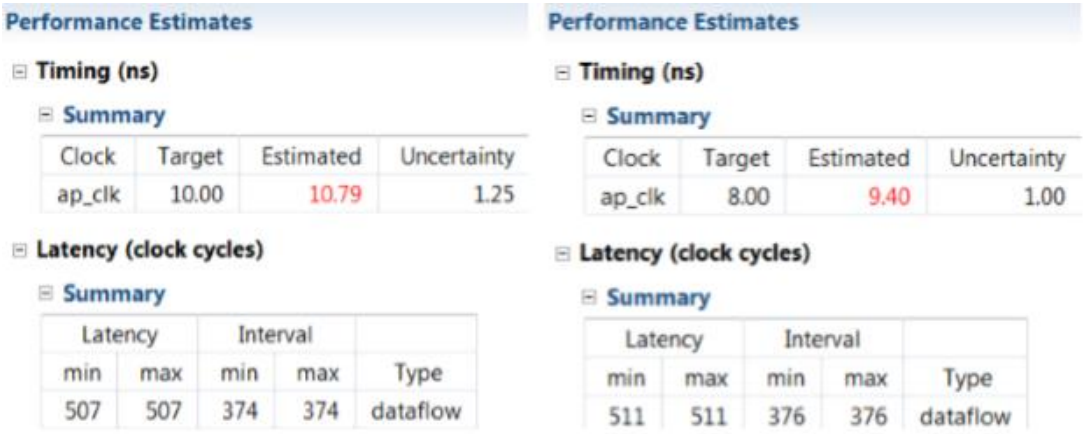
7.1.4. Kliknite pravým tlačidlom myši na funkciu **dct** v okne Directive a vyberte možnosť *Insert Directive...*

7.1.5. Vyberte direktívu **DATAFLOW** pre zlepšenie priepustnosti.

7.1.6. Spusťte syntézu.

7.1.7. Po skončení syntézy sa automaticky otvorí výsledná správa.

7.1.8. Všimnite si, že ako typ priepustnosti, ktorý je uvedený v Performance Estimates je dafalow.



Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	10.79	1.25

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
507	507	374	374	dataflow

a) ZedBoard

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	8.00	9.40	1.00

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
511	511	376	376	dataflow

b) Zybo

Obrázok 16 Porovnanie výkonu po použití direktívy DATAFLOW

Priepustnosť dataflow udáva počet taktových periód medzi každou množinou načítaných vstupov. Ak táto hodnota je menšia ako udáva latencia návrhu, návrh môže začať spracovávať nové vstupy pred tým, ako sa vstupy stanú výstupmi.
Všimnite si, že len dataflow je podporovaný pre funkcie a slučky na najvyššej úrovni, nie tie, ktoré sú na spodku hierarchie. Jedine slučky a funkcie, ktoré sú v rámci návrhu najvyššie získali výhody z optimalizácie dataflow.

7.1.9. Presuňte sa nižšie do oblasti Estimates, pozorujte, že hodnota BRAM_18K sa zvýšila z 3 na 4.

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	1
FIFO	-	-	-	-
Instance	2	8	985	572
Memory	2	-	512	32
Multiplexer	-	-	-	16
Register	-	-	12	-
Total	4	8	1509	621
Available	280	220	106400	53200
Utilization (%)	1	3	1	1

a) ZedBoard

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	1
FIFO	-	-	-	-
Instance	2	8	1026	576
Memory	2	-	512	32
Multiplexer	-	-	-	16
Register	-	-	12	-
Total	4	8	1550	625
Available	120	80	35200	17600
Utilization (%)	3	10	4	3

b) Zybo

Obrázok 17 Odhad zdrojov po použití direktívy DATAFLOW

7.1.10. V konzolovom okne si všimnite , že dct_coeff_table je automaticky rozdelená na 2 dimenzie. Polia buf_2d_in a col_inbuf sú rozdelené podľa pridelenej direktívy z predchádzajúceho behu. Dataflow je použité na najvyššej úrovni, vytvorilo kanály medzi funkciami read_data, dct_2d a write_data.

7.2. **Vykonanie analýzy výkonu prepnutím sa do pohľadu analýzy a prezeranie profilu výkonu dct.**

7.2.1. Prepnite sa do pohľadu analýzy, rozbaľte položky v Module Hierarchy a vyberte položku **dct_2d**.

7.2.2. Vyberte kartu Performance Profile.

Pozorujte, že najvyššia latencia a interval (priepustnosť) je spôsobená funkciou dct_2d. Interval funkcie najvyššej úrovne dct je nižší ako súčet intervalov funkcií read_data, dct_2d a write_data čo naznačuje, že pôsobia súčasne a dct_2d je obmedzujúci faktor. Na karte Performance Profile je možné vidieť, že dct_2d nie je kompletne vykonávaná paralelne ako Row_DCT_Loop a Col_DCT_Loop (tie boli prepojené).

Jedným z obmedzení optimalizácie dataflow je, že pracuje len na najvyššej úrovni slučiek a funkcií. Jeden spôsob, ako mať bloky v dct_2d prevádzkované paralelne, je prepojiť celú funkciu. To by však rozvinulo všetky slučky a niekedy to môže spôsobiť nárast veľkej plochy. Alternatívou je povýšiť tieto slučky na najvyššiu

úroveň hierarchie, kde je možné použiť dataflow optimalizáciu odstránením dct_2d hierarchie, t.j. vnoriť funkciu dct_2d.

7.2.3. Prepnite sa do perspektívy syntézy.

8. Použitie direktívy INLINE

8.1. Vyberte **Project > New Solution.**

8.2. V otvorenom dialógovom okne kliknite na **Finish** (s výberom Solution5).

8.3. Pravým tlačidlom myši kliknite na **dct_2d** na karte Directive a vyberte možnosť *Insert Directive...*

8.4. Z konceptového menu vyberte direktívu **INLINE**. Táto direktíva spôsobí, že sa hierarchia nerozpustí.

8.5. Spusťte syntézu.

8.6. Po skončení syntézy sa otvorí výsledná správa.

8.7. Všimnite si, že sa latencia znížila z 507 na 479 taktových períód pre ZedBoard (z 511 na 483 pre Zybo) a priepustnosť Dataflow sa výrazne znížila z 374 na 106 taktových períód pre ZedBoard (z 376 na 106 pre Zybo).

8.8. Prezrite si záznam syntézy aby ste videli, ktoré transformácie boli použité automaticky.

- Funkcia dct_1d je automaticky vnorená do slučky, z ktorej bola volaná. Tá umožňuje slučke vnáranie a bude automaticky zlúčená.
- Tiež si všimnite, že využitie DSP48E sa zdvojnásobilo (z 8 na 16). Je to preto, že predtým bola použitá jedna inštancia dct_1d pre tvorbu aj riadkov aj stĺpcov. Teraz sa riadky a stĺpce vykonávajú súčasne.
- Použitie BRAM sa opäť zvýšilo (z 4 na 6).

8.9. Prepnite sa do perspektívy analýzy, rozbaľte položky Module Hierarchy a vyberte položku **dct**. Pozorujte, že položka dct_2d je teraz nahradená

dct_Loop_Row_DCT_Loop_proc, dct_Loop_Xpose_Row_Outer_Loop_proc, dct_Loop_Col_DCT_Loop_proc a dct_Loop_Xpose_Col_Outer_Loop_proc pretože funkcia dct_2d je vnorená. Taktiež si všimnite, že všetky funkcie pracujú paralelne, prínosom je interval (priepustnosť) funkcie najvyššej úrovne predstavujúci 106 taktových periód.

8.10. Prepnite sa do perspektívy syntézy.

9. Použitie direktívy RESHAPE

9.1. Vyberte **Project > New Solution**.

9.2. V otvorenom dialógovom okne kliknite na **Finish** (s výberom Solution6).

9.3. Kliknite pravým tlačidlom myši na pole **buf_2d_in**, vyberte možnosť **Modify Directive** a zmeňte direktívu **PARTITION** na **ARRAY_RESHAPE** s hodnou dimenzie 2 a kliknite na **OK**.

9.4. Rovnako zmeňte direktívu poľa **col_inbuf** na **ARRAY_RESHAPE** s dimenziou 2.

9.5. Použite direktívu **ARRAY_RESHAPE** s hodnotou dimenzie 2 na **dct_coeff_table.**

9.6. Spusťte syntézu.

9.7. Po skončení syntézy sa automaticky otvorí výsledná správa.

9.8. Všimnite si, že obe latencie (zvýšené z 479 na 607 pre ZedBoard a z 483 na 611 pre Zybo) a dataflow priepustnosť (zvýšené zo 106 na 131 pre ZedBoard a zi 106 na 132 pre Zybo) klesli. Využitie prostriedkov BRAM vzrástli z 6 ma 22 pre ZedBoard aj Zybo.

- Záznam syntézy poskytuje niekoľko stôp. Sú tu upozornenia vo fáze plánovania pre read_data o tom, že II = 1 nie je možné dosiahnuť. V skutočnosti sa read_data sťažuje na operácie read a write.
- Táto operácia negatívne ovplyvňuje maximálnu šírku pre takéto pole.

9.9. Teda môžete vidieť, že direktívy boli použité starostlivo.

9.10. Zavrite Vivado HLS výberom **File > Exit**.

Odpovede

Odhadovaná perióda taktu:	6.38 ns
Najhorší prípad latencie:	3959 taktových období
Celková hodnota DSP48E:	1
Celková hodnota BRAM:	5
Celková hodnota FF:	278
Celková hodnota LUT:	354