

# **Construindo aceleradores em SYCL para computação de alto desempenho em FPGAs**

---



Prof. Ricardo Menotti ([menotti@ufscar.br](mailto:menotti@ufscar.br))

17 de setembro de 2020

## **XXI Simpósio em Sistemas Computacionais de Alto Desempenho**

**Departamento de Computação**

Centro de Ciências Exatas e de Tecnologia

Universidade Federal de São Carlos

# Roteiro

---

## 1. Introdução

Computação Heterogênea

## 2. FPGAs

Computação Reconfigurável

## 3. SYCL

DPC++

## 4. Práticas

Laboratórios

# Roteiro

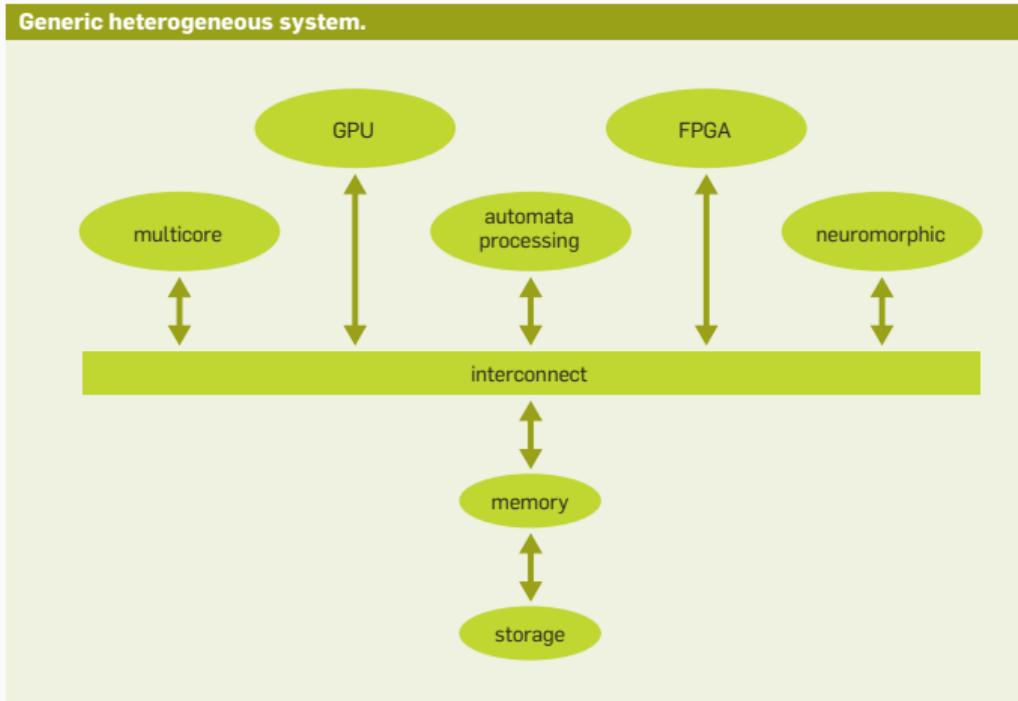
---

1. Introdução
  - Computação Heterogênea
2. FPGAs
  - Computação Reconfigurável
3. SYCL
  - DPC++
4. Práticas
  - Laboratórios

# Introdução

---

# Heterogeneous Computing: Here to Stay (Zahran, 2017)

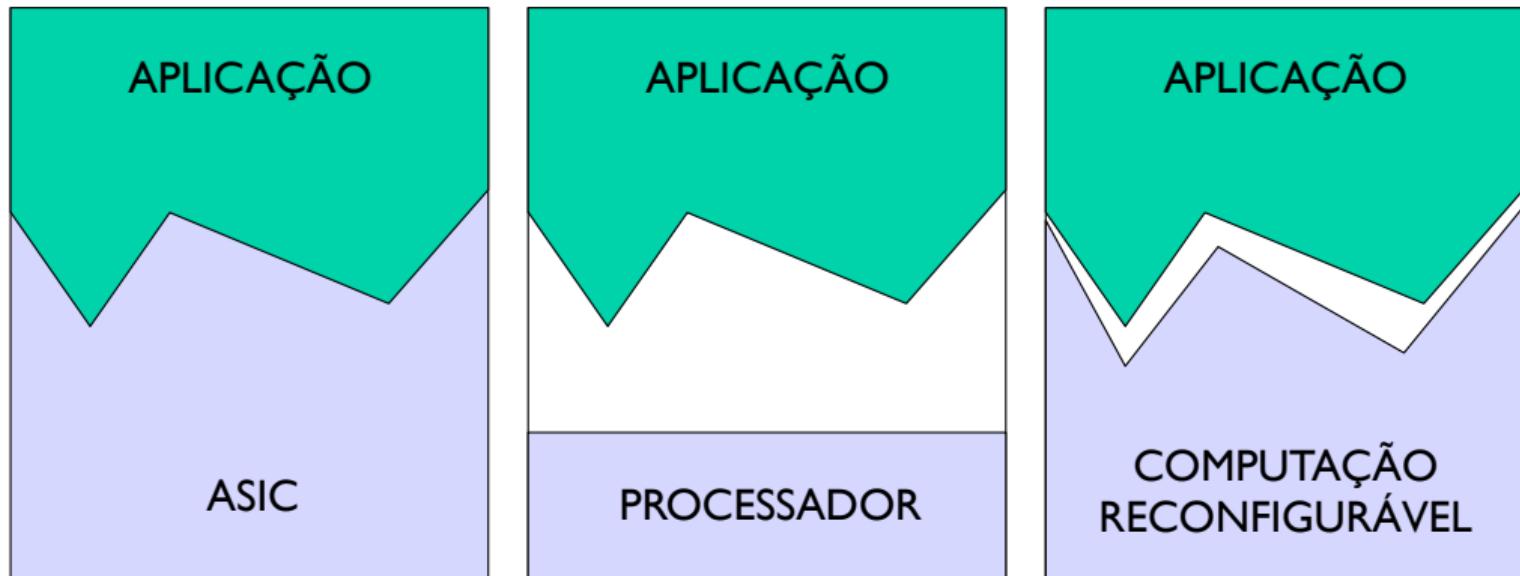


1. Núcleos idênticos ... tensão dinâmica e escala de frequência;
2. Núcleos com características arquiteturais distintas que parecem executar instruções em sequência, ...;
3. Nós de computação com diferentes modelos de execução, tais como GPUs, que processam múltiplos dados com instrução única (ou *thread*);
4. Aceleradores programáveis (FPGAs) que podem ser usados como hardware especializado...

# FPGAs

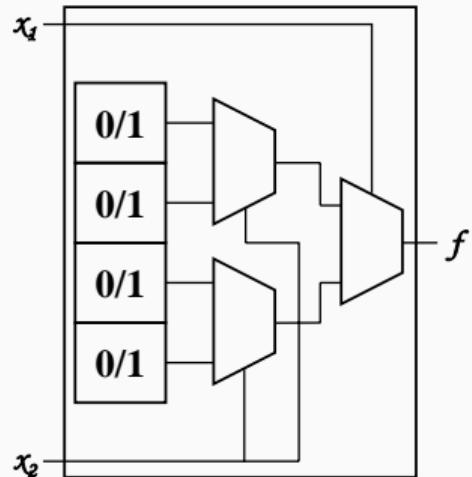
---

# Computação Reconfigurável



**Figura 1:** Computação reconfigurável comparada às soluções de *hardware dedicado* (ASIC) e *software* executando em processador de propósito geral (Menotti, 2010)

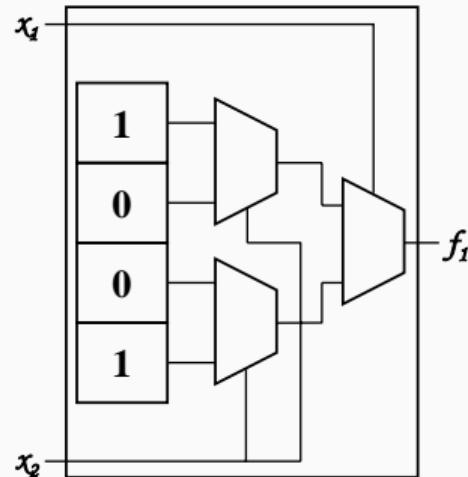
## Look-Up Tables (LUTs)



(a)

$x_1$	$x_2$	$f_1$
0	0	1
0	1	0
1	0	0
1	1	1

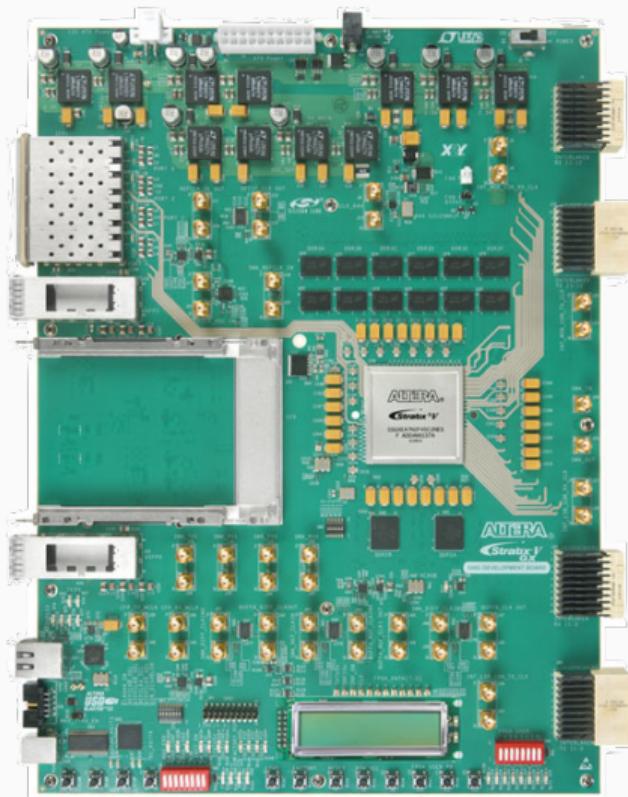
(b)



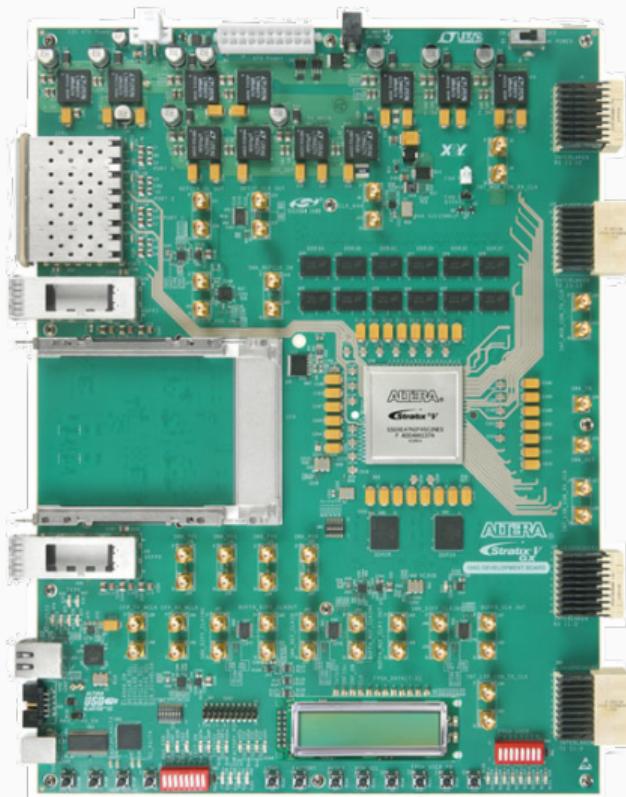
(c)

**Figura 2:** Implementação de uma função lógica em uma LUT (Menotti, 2010)

# Kits ou Placas FPGAs



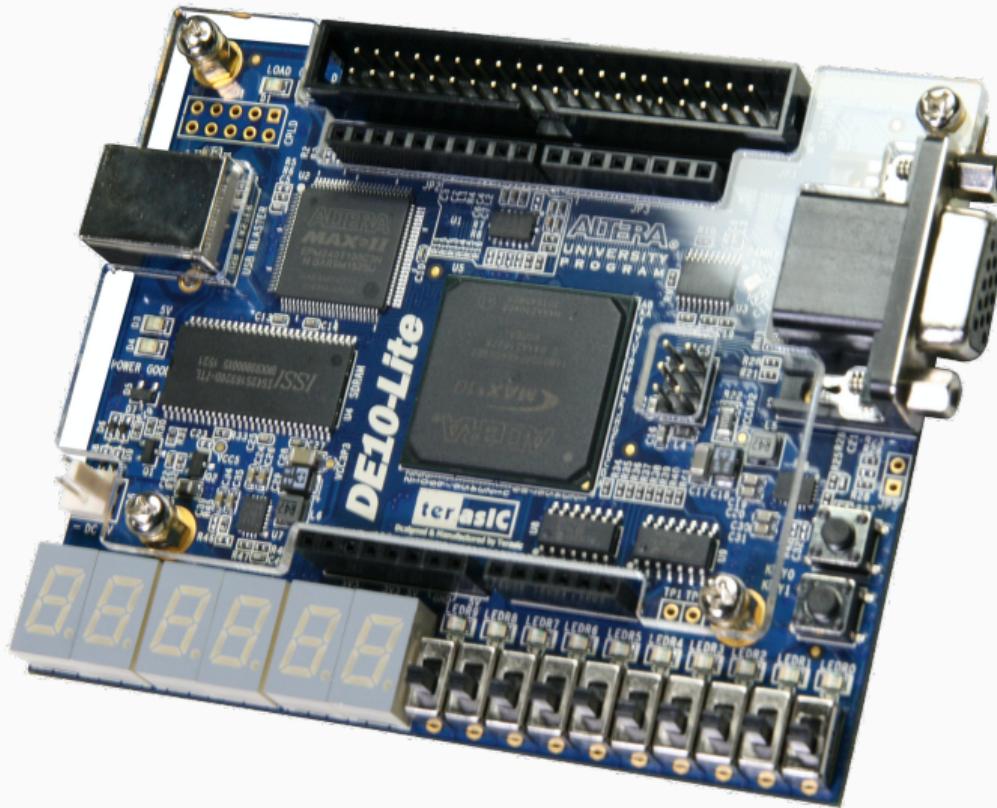
## Kits ou Placas FPGAs



## Kits ou Placas FPGAs



# Kits ou Placas FPGAs



# Fluxo de desenvolvimento para FPGAs

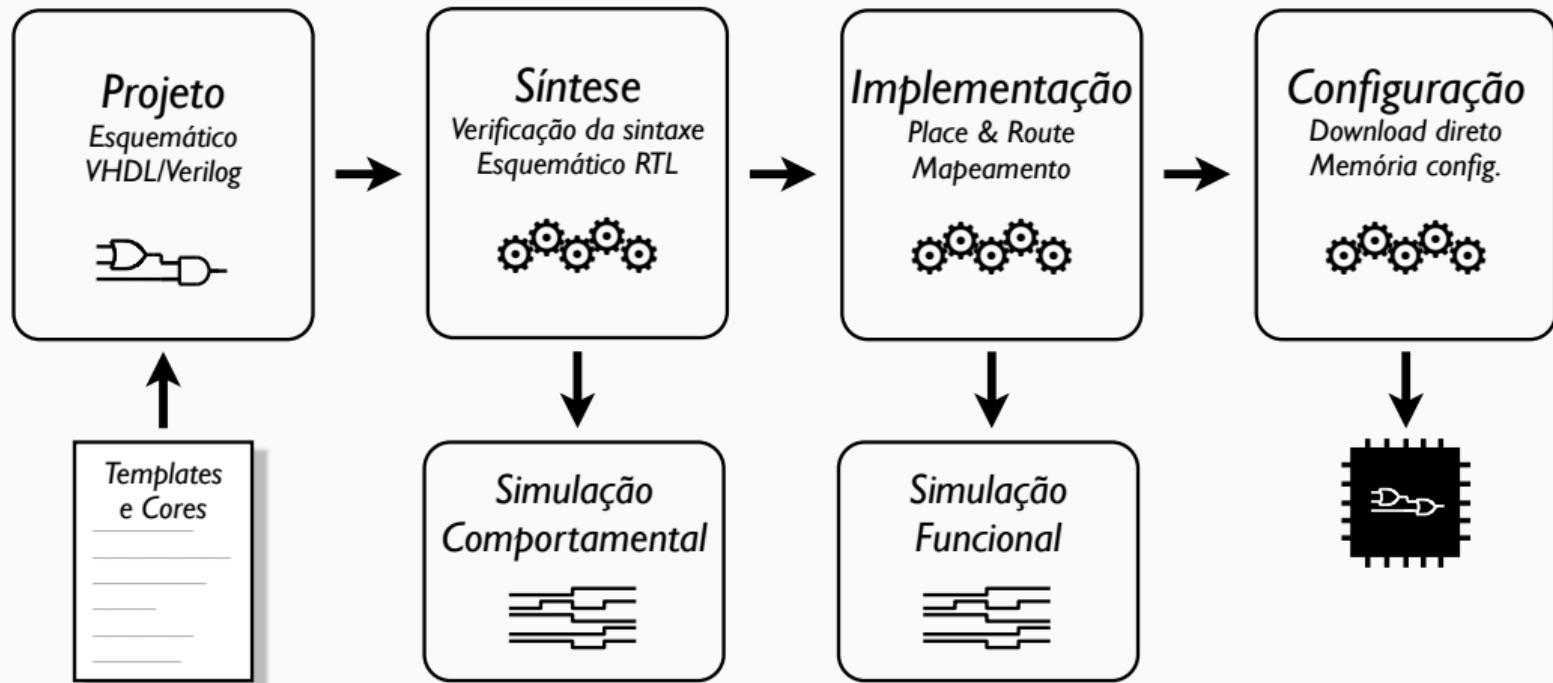
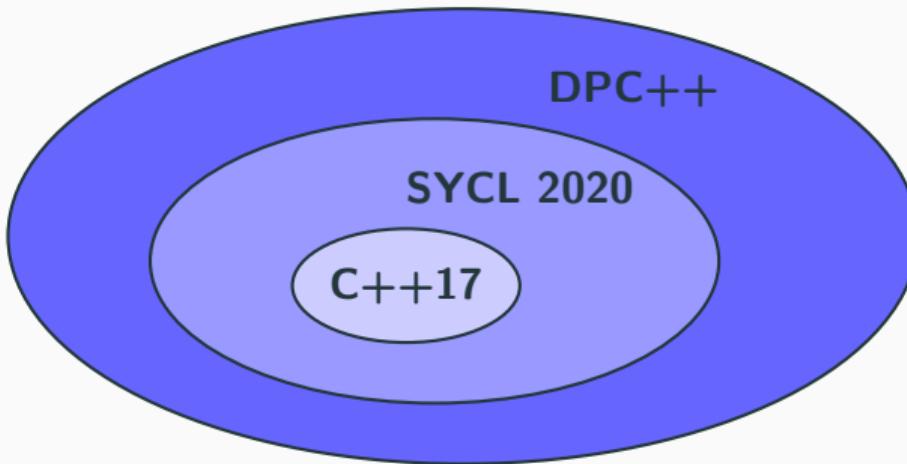


Figura 3: Fluxo de desenvolvimento para FPGAs (Menotti, 2010)

**SYCL**

---



**Figura 4:** Relação de DPC++ com SYCL e C++ (Intel Corp., 2020)

# Templates

```
1 #include <iostream>
2
3 template <class T>
4 T GetMax (T a, T b) {
5     T result;
6     result = (a > b) ? a : b;
7     return (result);
8 }
9
10 int main () {
11     int i = 7, j = 5, k;
12     float l = 1.0, m = 5.5, n;
13     k = GetMax<int>(i, j);
14     n = GetMax<float>(l, m);
15     std::cout << k << std::endl;
16     std::cout << n << std::endl;
17     return 0;
18 }
```

*São funções/classes especiais que podem operar com tipos genéricos, cuja funcionalidade pode ser adaptada a mais de um tipo ou classe sem repetir o código inteiro para cada tipo.*

Saída:

7

5.5

Experimente este exemplo mais complexo depois...

# Lambdas

---

```
1 #include <iostream>
2 #include <typeinfo>
3
4 int main()
5 {
6     int i = 1;
7     int j = 2;
8
9     auto f = [&i, j](int a) -> double
10    { return i + j + a; };
11
12    i = 4;
13    j = 8;
14
15    std::cout << f(1)  << std::endl;
16    std::cout << typeid(f(42)).name()
17    << std::endl;
18 }
```

*São uma maneira conveniente de definir um objeto de função anônimo.*

Saída:

7  
d

Modifique este exemplo depois...

## Inferência de tipos (auto)

```
1 std::vector<int> v {2, 7, 42};  
2 for (std::vector<int>::iterator e = v.begin(); e != v.end(); ++e)  
3     std::cout << *e << std::endl;
```

## Inferência de tipos (auto)

```
1 std::vector<int> v {2, 7, 42};  
2 for (std::vector<int>::iterator e = v.begin(); e != v.end(); ++e)  
3     std::cout << *e << std::endl;
```

```
1 std::vector v {2, 7, 42};  
2 for (auto e : v)  
3     std::cout << e << std::endl;
```

## Inferência de tipos (auto)

```
1 std::vector<int> v {2, 7, 42};  
2 for (std::vector<int>::iterator e = v.begin(); e != v.end(); ++e)  
3     std::cout << *e << std::endl;
```

```
1 std::vector v {2, 7, 42};  
2 for (auto e : v)  
3     std::cout << e << std::endl;
```

```
1 // Desafio: tente especificar o tipo da variável acc  
2 auto acc = buf.get_access<sycl::access::mode::read_write>(cgh);
```

## Práticas

---

# Mãos à obra!

1. Acesse o Jupyter!
2. Abra um terminal (Inception!)
3. `git clone https://github.com/menotti/sycl-fpga-wscad-2020`

Obrigado!

# **Construindo aceleradores em SYCL para computação de alto desempenho em FPGAs**

---



Prof. Ricardo Menotti ([menotti@ufscar.br](mailto:menotti@ufscar.br))

17 de setembro de 2020

## **XXI Simpósio em Sistemas Computacionais de Alto Desempenho**

**Departamento de Computação**

Centro de Ciências Exatas e de Tecnologia

Universidade Federal de São Carlos

# Referências

---

-  Intel Corp. (2020). *FPGA Add-On for oneAPI Base Toolkit(Beta)*.  
<https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/fpga.html>
-  Menotti, R. (2010). *LALP: uma linguagem para exploração do paralelismo de loops em computação reconfigurável* (tese de dout.). Universidade de São Paulo.  
<https://doi.org/10.11606/T.55.2010.tde-17082010-151100>
-  Zahran, M. (2017). Heterogeneous Computing: Here to Stay. *Commun. ACM*, 60(3), 42–45. <https://doi.org/10.1145/3024918>