V8 call interface descriptors

软件所智能软件中心PLCT实验室 邹小芳

2020/03/19

目录

01 What is call interface descriptors

V8中描述函数参数及返回值信息的classes, in src/codegen/interface-descriptors.[h|cc]

For builtins

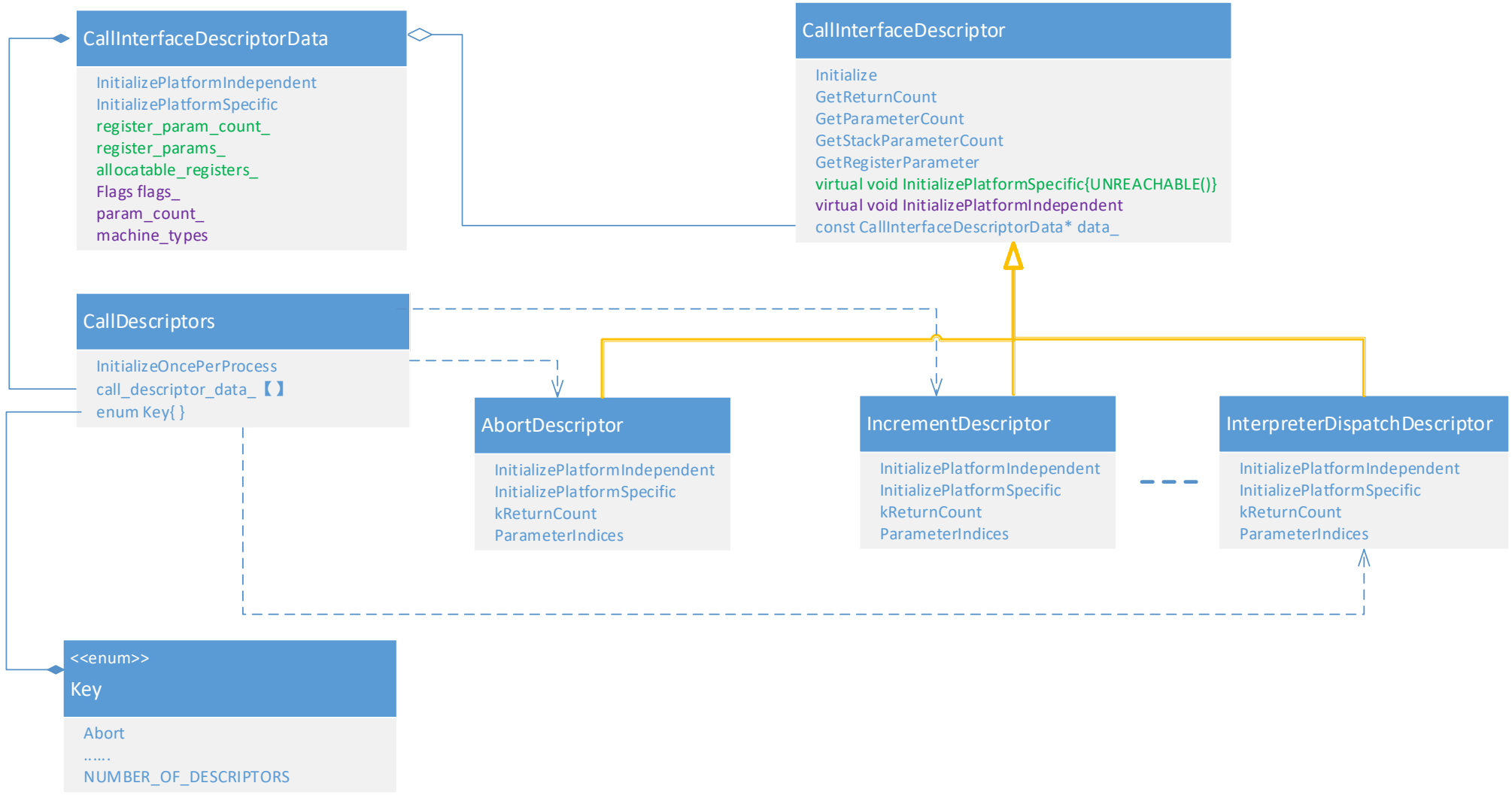
- **TFC: Builtin in Turbofan, with CodeStub linkage.**
  Args: name, interface descriptor
  example: TFC(Abort, Abort)
- **TFH: Handlers in Turbofan, with CodeStub linkage.**
  Args: name, interface descriptor
  example: TFH(LoadIC_StringLength, LoadWithVector)
- **TFS: Builtin in Turbofan, with CodeStub linkage.**
  Args: name, explicit argument names...
  example: TFS(Increment, kValue)
- **BCH: Bytecode Handlers, with bytecode dispatch linkage.**
  Args: name, OperandScale, Bytecode
  example: BCH(AddHandler, interpreter::OperandScale::kSingle, interpreter::Bytecode::kAdd)

```
class CallDescriptors : public AllStatic
{
  public:
 enum Key {
    Abort,
    ...
    LoadWithVector,
    Increment,
    InterpreterDispatch,
    ...
    NUMBER_OF_DESCRIPTORS
}
static void InitializeOncePerProcess();
private:
  static CallInterfaceDescriptorData
call_descriptor_data_[NUMBER_OF_DESCRIPTORS];
}
```

TFC(Add, BinaryOp)
TFC(Subtract, BinaryOp)
} 多个builtins可以共用一个descriptor

01 what is call interface descriptors

Class diagram



02 Initialize call interface descriptors

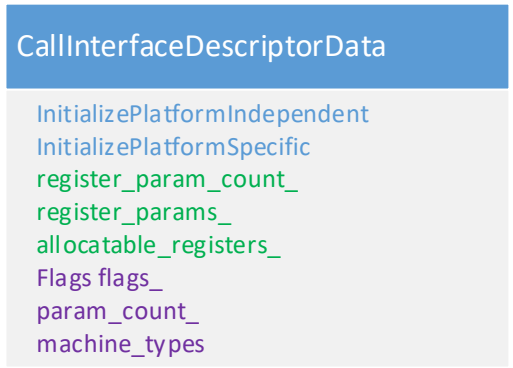
V8 Initialize

```
bool V8::Initialize() {
  InitializeOncePerProcess();
  return true;
}
void V8::InitializeOncePerProcess() {
  base::CallOnce(&init_once, &InitializeOncePerProcessImpl);
}

void V8::InitializeOncePerProcessImpl() {
.....
 Bootstrapper::InitializeOncePerProcess();
 CallDescriptors::InitializeOncePerProcess();
 wasm::WasmEngine::InitializeOncePerProcess();
}
```

02 Initialize call interface descriptors

Call interface descriptors initialize

```
void CallDescriptors::InitializeOncePerProcess() {
#define INTERFACE_DESCRIPTOR(name, ...) \
  name##Descriptor().Initialize(&call_descriptor_data_[CallDescriptors::name]);
  INTERFACE_DESCRIPTOR_LIST(INTERFACE_DESCRIPTOR)
#undef INTERFACE_DESCRIPTOR
...
}

void CallInterfaceDescriptor::Initialize(CallInterfaceDescriptorData* data) {
    InitializePlatformSpecific(data);
    InitializePlatformIndependent(data);
}
```

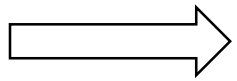


03 Use call interface descriptors

CallDescriptor

```
CallDescriptor* Linkage::GetStubCallDescriptor(
    Zone* zone, const CallInterfaceDescriptor& descriptor,
    int stack_parameter_count, CallDescriptor::Flags ,
    Operator::Properties , StubCallMode stub_mode) {
    ......
    new (zone) CallDescriptor(....);
}
```

```
// example 1: RawMachineAssembler::CallN
Node* RawMachineAssembler::CallN(CallDescriptor* call_descriptor,       int
input_count, Node* const* inputs) {
  DCHECK(!call_descriptor->NeedsFrameState());
  DCHECK_EQ(input_count, call_descriptor->ParameterCount() + 1);
  return AddNode(common()->Call(call_descriptor), input_count, inputs);
}
```



CallDescriptor

- kind
- IsCFunctionCall
- IsJSFunctionCall
- IsWasmFunctionCall
- IsWasmImportWrapper
- IsWasmCapiFunction
- ReturnCount
- ParameterCount
- StackParameterCount
- StackReturnCount
- GetReturnLocation
- GetInputLocation
- GetReturnType
- GetInputType
- CalleeSavedFPRegisters
- CalleeSavedRegisters
- AllocatableRegisters
- UsesOnlyRegisters
- GetParameterType
- target_type_
- kind_
- target_loc_
- stack_param_count_
- stack_return_count_
- callee_saved_registers_
- callee_saved_fp_registers_
- allocatable_registers_
- flags_

03 Use call interface descriptors

Example

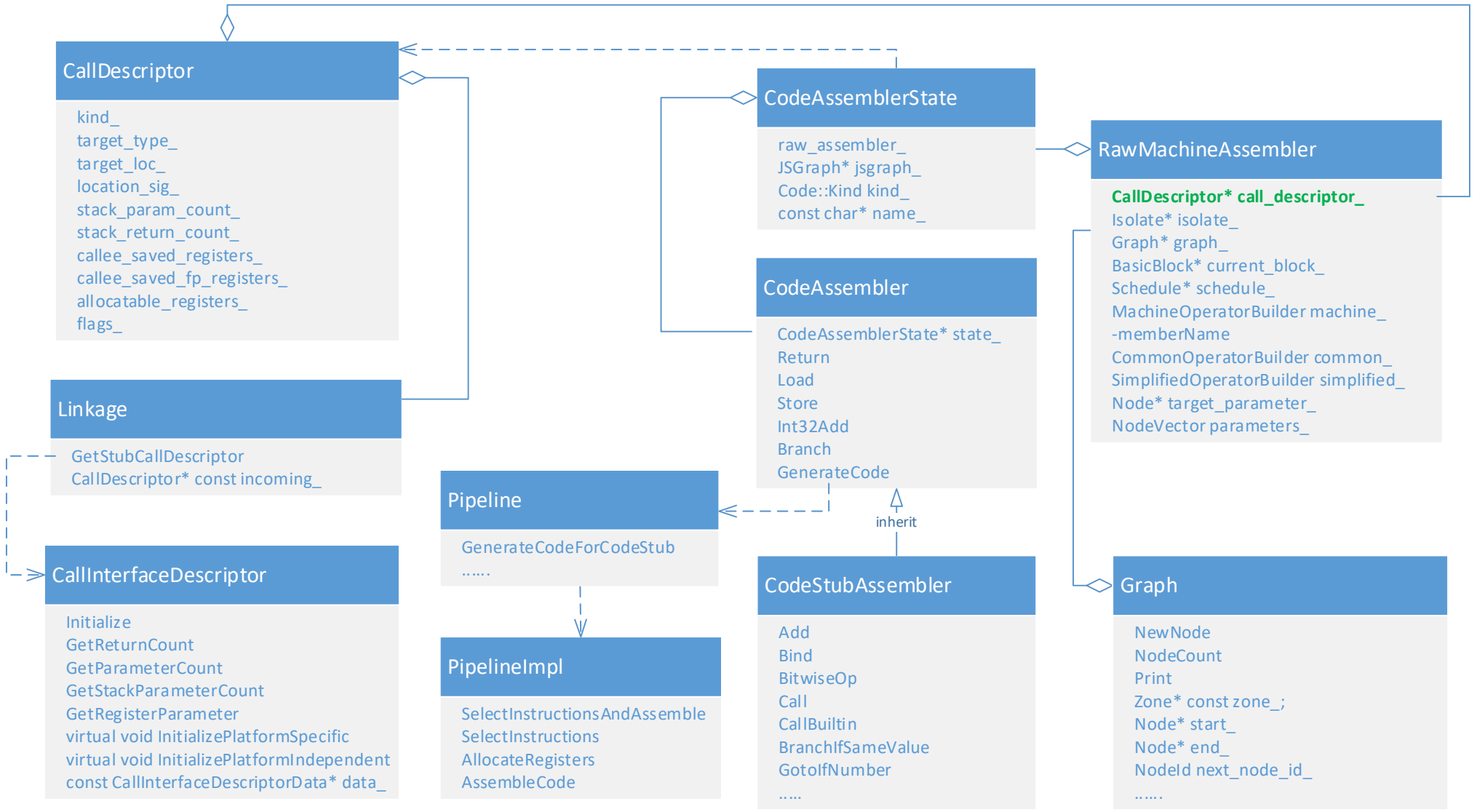
```
// example 2:  InstructionSelector::VisitReturn
void InstructionSelector::VisitReturn(Node* ret) {
  OperandGenerator g(this);
  const int input_count = linkage()->GetIncomingDescriptor()->ReturnCount() == 0  ? 1   :
ret->op()->ValueInputCount();
......
Emit(kArchRet, 0, nullptr, input_count, value_locations);
}
```

```
// example 3: CodeGenerator::AssembleReturn
void CodeGenerator::AssembleReturn(InstructionOperand* pop) {
  auto call_descriptor = linkage()->GetIncomingDescriptor();
  const RegList saves = call_descriptor->CalleeSavedRegisters();
......
}
```

```
CallDescriptor* GetIncomingDescriptor() const { return incoming_; }
```

03 Use call interface descriptors

Class diagram



04 To do for new architecture

Implement interface-descriptors-xxx.cc for new architecture

- **Implement funciton “InitializePlatformSpecific”**

Some classes inherited from CallInterfaceDescriptor haven't provided an implementation in interface-descriptors.[h|cc], they should be implemented it in file interface-descriptors-xxx.cc

- **Implement auxiliary function needed**

Some functions are declared but undefined in interface-descriptors.[h|cc], they should be implemented it in file interface-descriptors-xxx.cc.

The functions to be implemented in interface-descriptors-xxx.cc are list here:

funcitons to be implemented

谢 谢

欢迎交流合作

2020/3/19