Beyond the PDP-11: Architectural support for a memory-safe C abstract machine

University of Cambridge: David Chisnall, Colin Rothwell, Robert N. M. Watson, Jonathan Woodruff, Munraj Vadera, Simon W. Moore, Michael Roe
SRI International: Brooks Davis, Peter G. Neumann
Presented by: Omri Mor

Why talk about the PDP-11?

- First target for C
- Byte-addressable memory
- Most contemporary computers use a similar model
- Little to no memory safety

Specification vs Implementation

- Memory-safety within the C specification is possible
- Software makes assumptions about the compiler and machine model
  - Commonly-implemented *undefined behavior*
  - Implementation-defined behavior

Terminology: *object*

C11 Standard section 3.15:

object: region of data storage in the execution environment, the contents of which can represent values

Common Idioms: Undefined Behavior

- Deconst: remove const qualifier from pointer
- II: *invalid intermediate*, pointer arithmetic in which an intermediate result is outside the bounds of an object
- Wide: storing a pointer in a smaller integer variable
- Last Word: accessing objects as aligned words when its extent is not word-aligned

Common Idioms: Machine Model Assumptions

- Container: access an enclosing structure via a pointer to a member
- Sub: arbitrary pointer subtraction
- Int: storing a pointer in an integer variable **in memory**
- IA: integer arithmetic on pointers (generalization of Int)
- Mask: masking of pointers (storing data in “extra” bits)

Example: The mask idiom

```
// The low bit of an aligned pointer is
// always 0, so we can hide a flag in it
int *set_flag(int *b)
{
   return (int*)((intptr_t)b | 1);
}
```

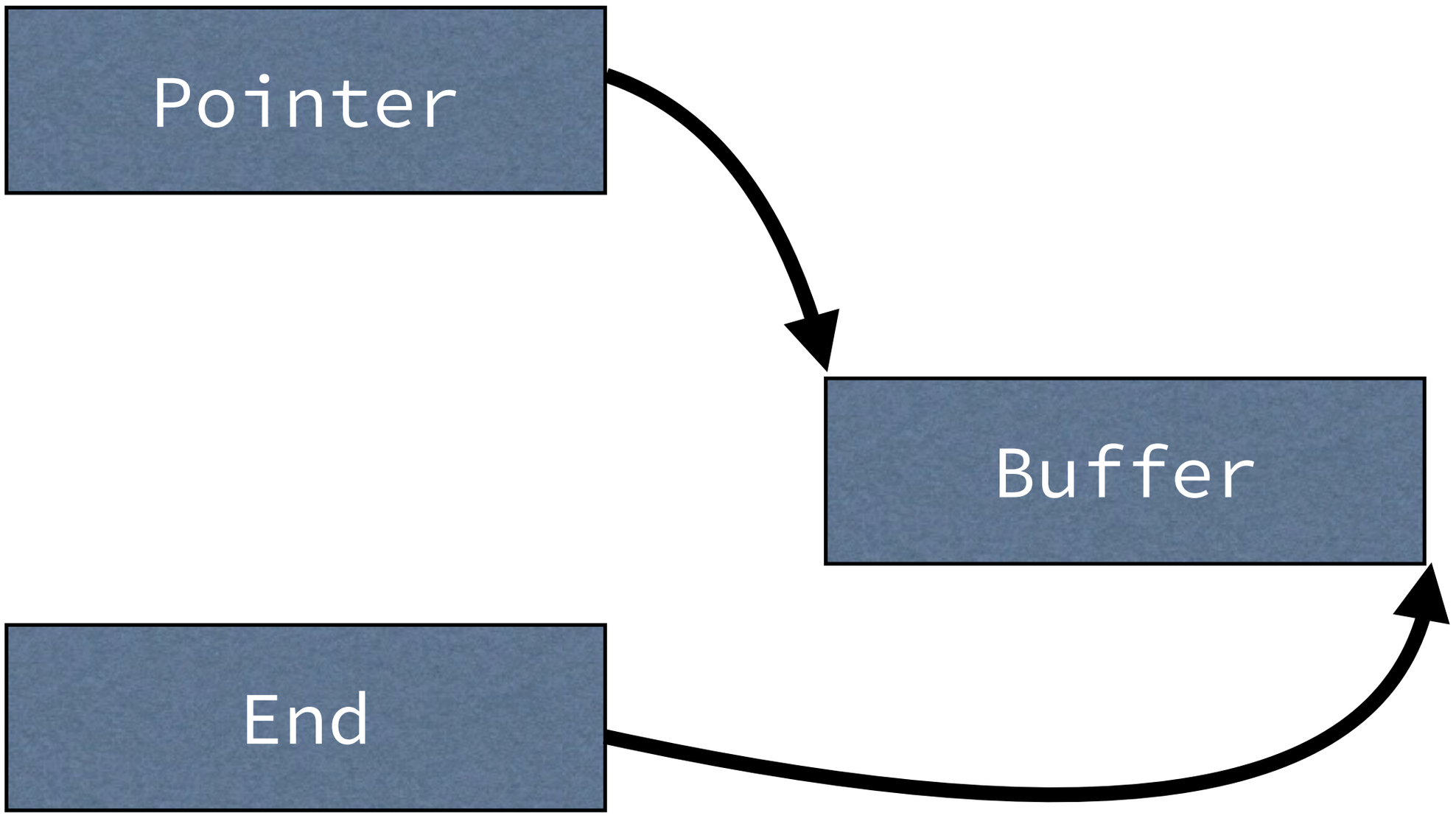
00x16012311230

Example: The mask idiom

```
// The low bit of an aligned pointer is
// always 0, so we can hide a flag in it
int *set_flag(int *b)
{
   return (int*)((intptr_t)b | 1);
}
```

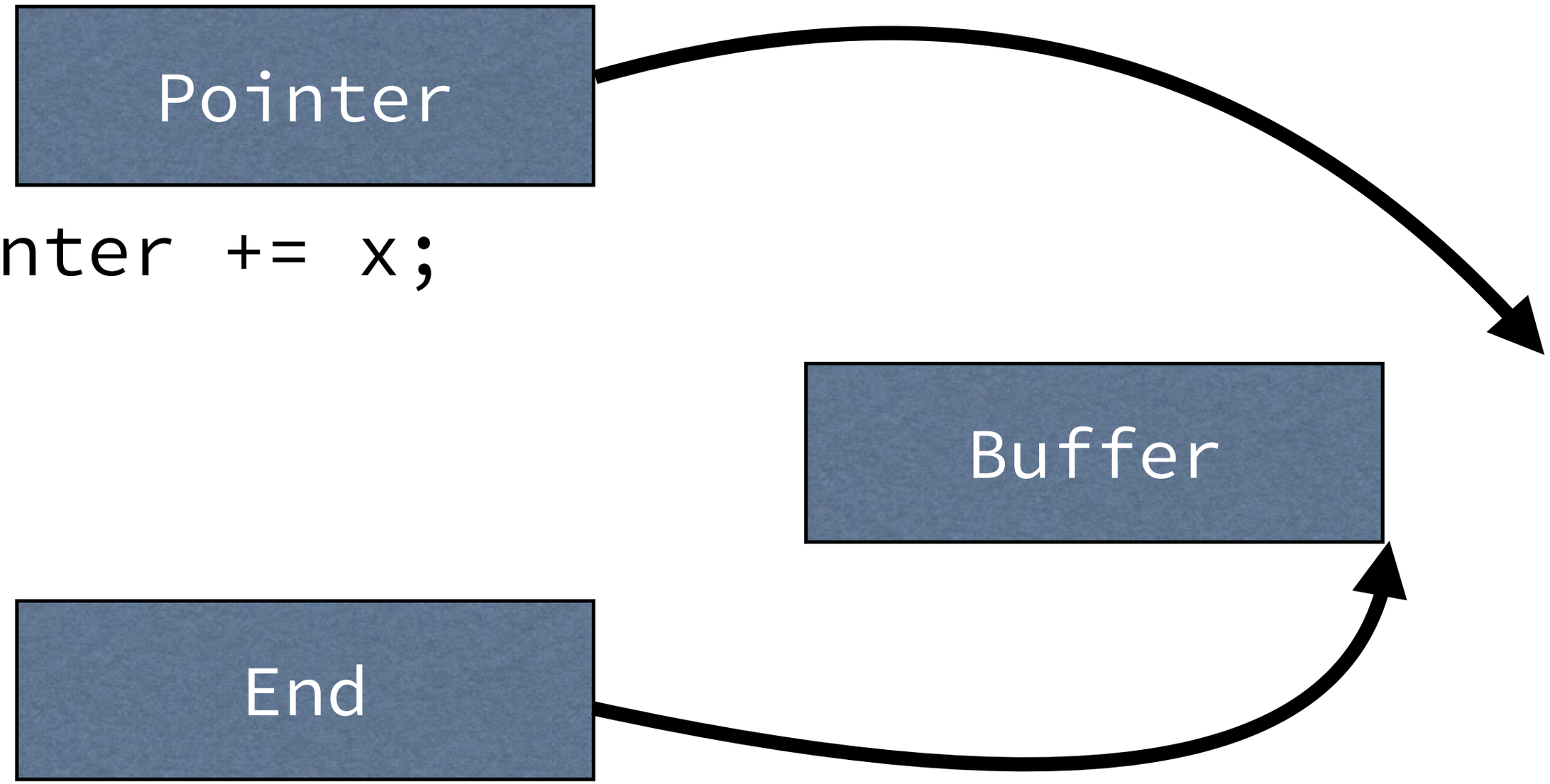
00x160123123I

Example: Invalid Intermediates



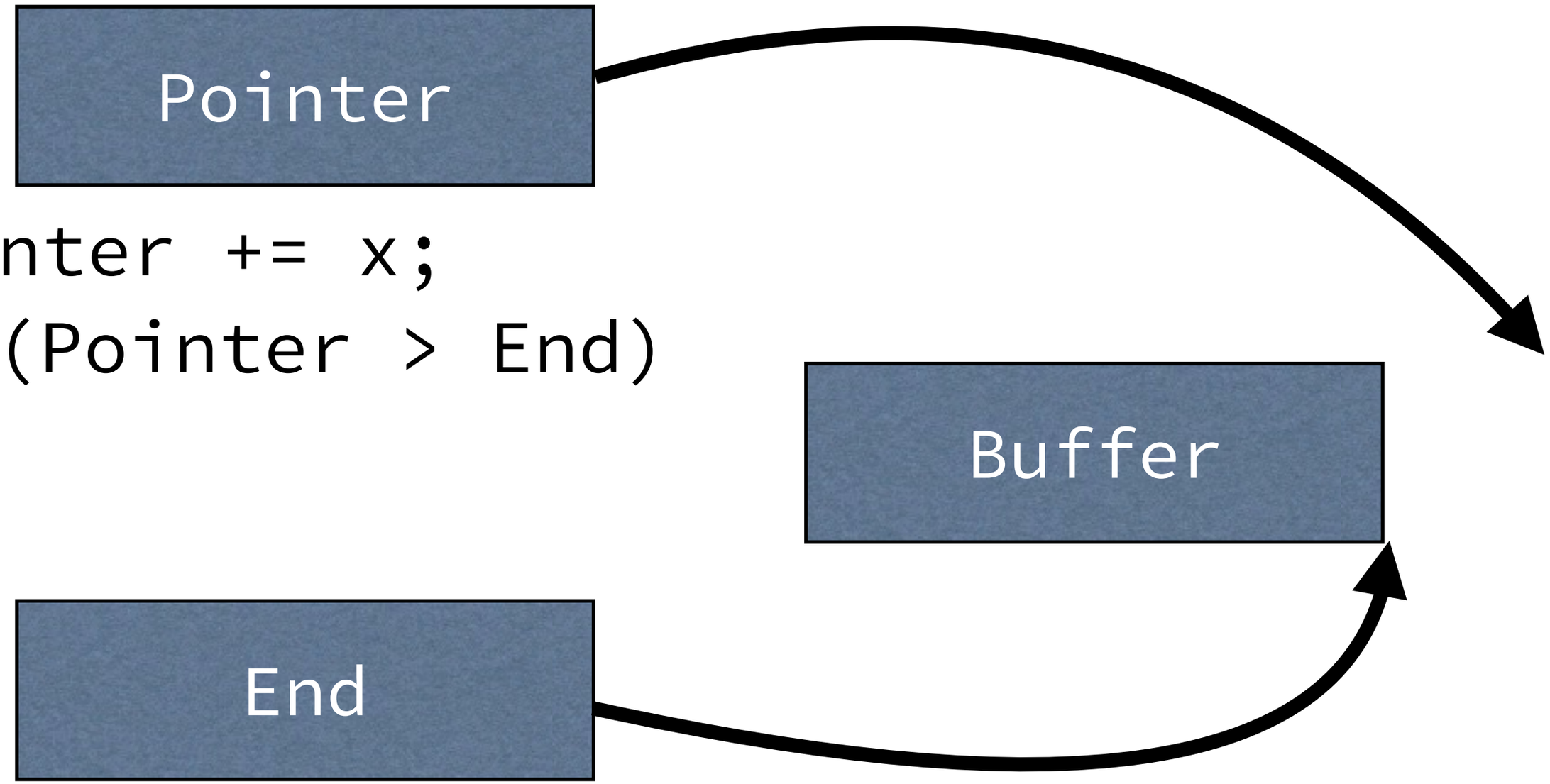
Example: Invalid Intermediates

```
Pointer += x;
```



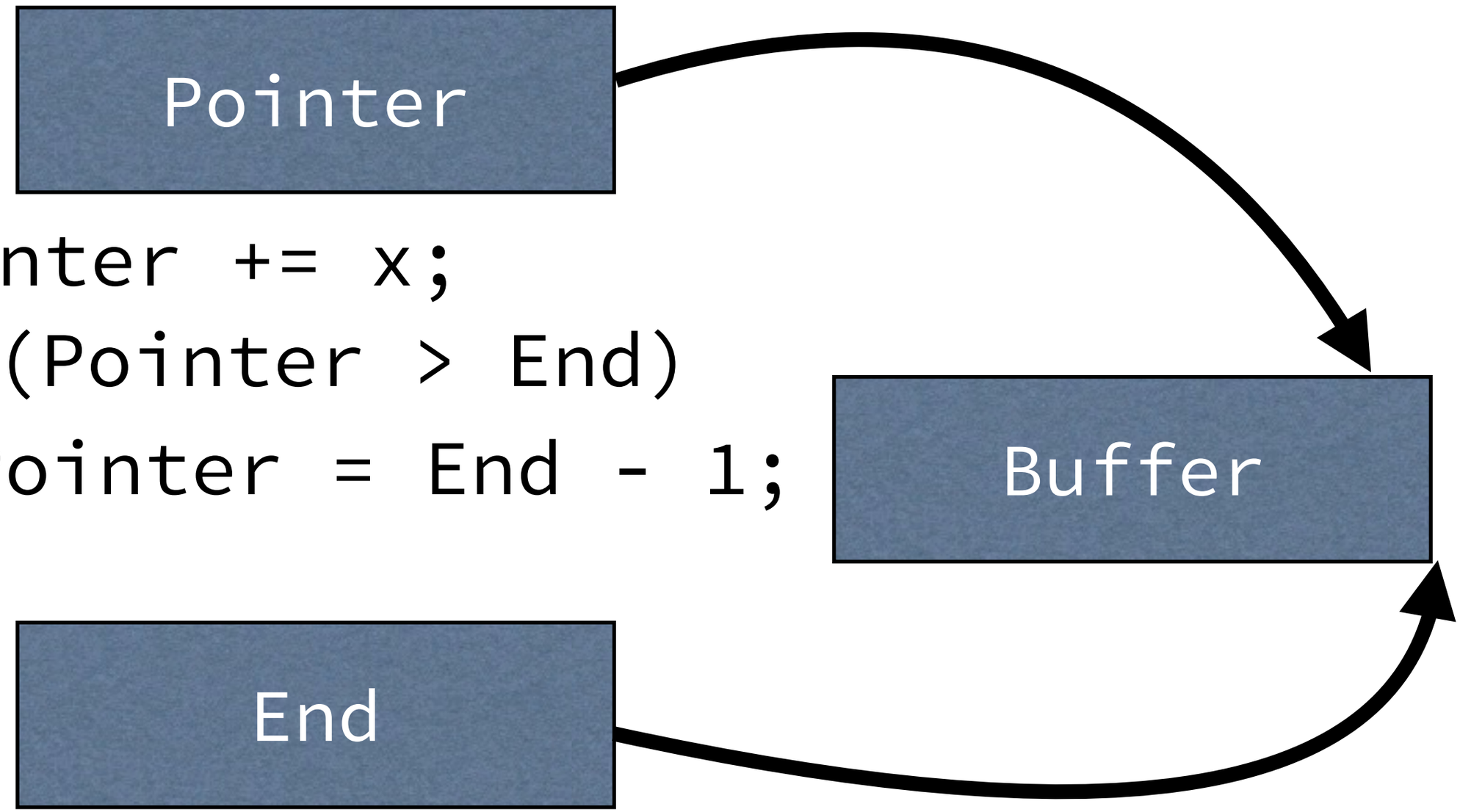
Example: Invalid Intermediates

```
Pointer += x;
if (Pointer > End)
```



Example: Invalid Intermediates

```
Pointer += x;
if (Pointer > End)
   Pointer = End - 1;
```



Statistics

Program	Deconst	Container	Sub	II	Int	IA	Mask	Wide	LoC
ffmpeg	150	0	800	4	0	0	4	0	693,010
libX11	117	0	19	9	1	0	0	5	120,386
FreeBSD libc	288	0	216	2	13	50	184	17	136,717
bash	43	0	207	11	0	0	15	4	109,250
libpng	20	0	175	1	0	0	0	0	50,071
tcpdump	579	0	9	1299	0	0	0	0	66,555
perf	575	151	46	0	53	151	31	4	52,033
pmc	2	0	0	0	18	0	0	0	8,886
pcre	98	0	52	0	0	0	0	0	70,447
python	494	0	358	1	109	0	131	8	383,813
wget	55	0	61	0	3	0	1	10	91,710
zlib	4	0	24	0	0	0	0	0	21,090
zsh	29	0	267	0	0	0	5	5	98,664
Total	2491	151	2236	1557	197	201	371	53	1,902,632

CHERI ISA

- 64-bit MIPS IV ISA
- Augmented with fine-grained *memory capabilities*
  - Represented as a *(base, bound, permissions)* triplet
- Memory accesses indirected via capabilities

Fat Pointers

- Extend CHERI capabilities
- Add offset to the triplet: *(base, bound, offset, permissions)*
- New instructions to manipulate the offset

Fat Pointers

Instruction	Use
CIncOffset	Adds an integer to the offset
CSetOffset	Sets the offset
CGetOffset	Returns the current offset
CPtrCmp	Compares two capabilities
CFromPtr	Converts a MIPS pointer to a capability
CToPtr	Converts capability to a MIPS pointer

New C Qualifiers

- Enforce const
  - __input
  - __output
- Add capabilities
  - __capability

C Implementation Models

- Traditional (x86/MIPS/PDP-11)
- HardBound: ASPLOS ’13, from UPenn
- Intel MPX: x86 bounds-checking extensions
- Relaxed Interpreter: pointers can be constructed from integers if object is valid
- Strict Interpreter: pointers can be constructed from integers if and only if they are unmodified
- CHERIv2: prior version of CHERI
- CHERIv3: CHERI with fat pointers

C Implementation Models

PROGRAM	DECONST	CONTAINER	SUB	II	INT	IA	MASK	WIDE
x86/MIPS/ PDP-11	yes	yes	yes	yes	yes	yes	yes	no
HardBound	yes	yes	yes	yes	(yes)	no	no	no
Intel MPX	yes	no	yes	yes	(yes)	(yes)	(yes)	no
Relaxed	yes	yes	yes	yes	yes	yes	yes	no
Strict	yes	yes	yes	yes	(yes)	no	no	no
CHERIv2	no	no	no	no	(yes)	no	no	no
CHERIv3	yes	yes	yes	yes	(yes)	yes	yes	no

Porting to CHERI

Program	Baseline LoC	CHERIv2			CHERIv3		
		Annotation	Semantic	Total	Annotation	Semantic	Total
Olden	1519	53 (3.5%)	0 (0%)	53 (3.5%)	53 (3.5%)	0 (0%)	53 (3.5%)
Dhrystone	448	11 (2.4%)	0 (0%)	11 (2.4%)	11 (2.4%)	0 (0%)	11 (2.4%)
tcpdump	66555	3006 (4.5%)	**1577 (2.4%)**	4583 (6.9%)	3006 (4.5%)	**2 (0.0%)**	3008 (4.5%)